TALKING ELECTRONICS®

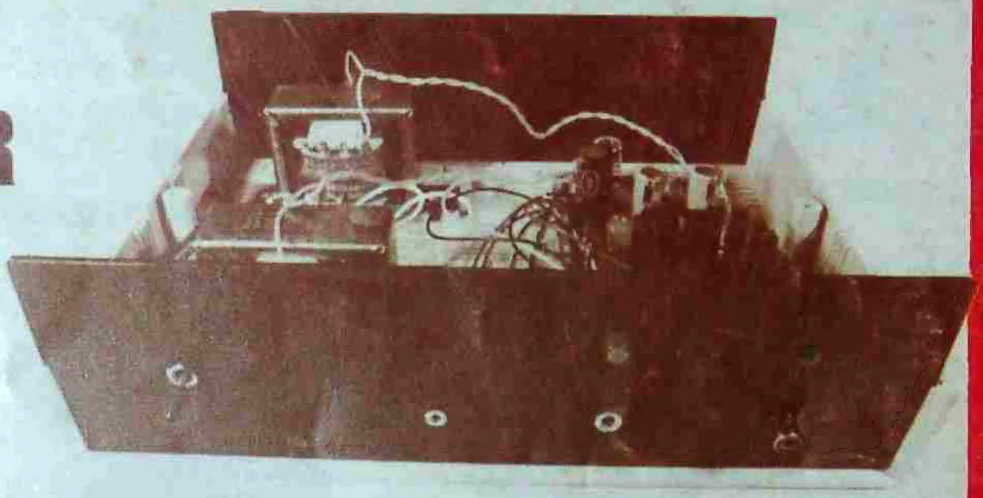
$2.20

$3.00NZ

Issue No 13

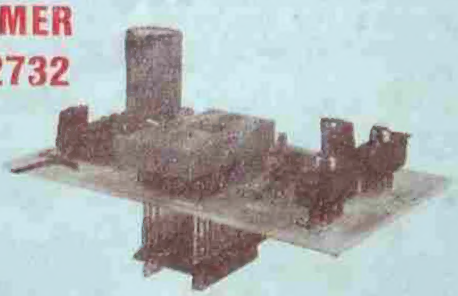
COMPUTER POWER SUPPLY

+5v FOR TTL
+12v FOR RELAYS
+30v FOR EPROM PROGRAMMING



2 PROGRAMMING AIDS

NON-VOLATILE RAM
EPROM PROGRAMMER
FOR 2716/2732



LOGIC PULSER

KITT SCANNER

20058520k

TALKING ELECTRONICS

Vol. 1 No: 13.

Editorial...

THE POWER OF SOFTWARE

Throughout our youth we are asked "What do you want to be?"
To reply Actor, or Designer, comes the inevitable "But what do you want to do for a REAL job?"
The only answer which satisfies is something concrete; something which the enquirer can relate to!
The same applies to electronics. If you answer Product Designer or Software Programmer many of the older generation will still want to know what you will be doing during working hours.
Up to now, areas like this, have been regarded as past-time endevours, like writing a book. If you said you wanted to be a Program Writer or Programmer, you were greeted with a muted "Oh well, another casual worker."
But now things have changed. Or let's hope so. The call for programming has taken on a new dimension. Whereas the hardware side of the computer market has taken an enormous dive, the software side is flourishing.

Software firms are going from strength to strength in their quest to fulfill orders both from Australia, United States and beyond.
The computer market has almost reached saturation and those with existing equipment are wanting to get more out of their investment.
The only way this can be achieved is with new and updated programs.
SOFTWARE PROGRAMS.
Programs which keep tighter control on stock and figures and even introduce new parameters.
The profitability of many businesses is getting so tight that the line between 'staying in business' and 'closing the doors' is getting finer and finer.
If we cite just two examples, you see what we mean.
Self Service petrol stations are currently operating on a gross margin of less than 3%, out of which the operator must pay running expenses and wages etc.
The same appies to liquor stores. Their nett profitability after paying expences, advertising and wages is between 3.5% and 5%.
In both these cases, a software programming firm has been able to increase the profitability of these operations by about 1%. This may not seem much but a rise of 1% accounts for an improvement of between 20 - 30%. The proof of the success of these programs is in their acceptance. Both businesses gladly paid some $2,000 for the improved program.
It effectively converted their out-of-date equipment into a daily stock-controller in which restocking was systematic and daily profitability figures were available.
In our own small way we have experienced the power of programming and this has been demonstrated in our computer projects.
It started with the TEC and continued with the MICROCOMP. Both these have shown us how a micro operating system can be developed as a basic unit and adapted to suit a wide range of specific situations.
The only need is to design a small amount of interfacing circuitry and the balance is achieved with a PROGRAM.
If you are following our series on the TEC you will very interested in the Microcomp. It is an extension to the TEC in that you write programs on the TEC and execute them on the 'comp.
If you are not quite up to this level, the programming hints covered in this issue will bring you one step closer.
And if you have not yet invested in the TEC computer, now is your opportunity to delve into the working of a micro system. The TEC is one of the simplest computers on the market and will get you started in programming.
Then, in a few years, when you are asked "What do you want to do" you can say 'Micro-Programmer' and they still won't be any wiser.

Colin Mitchell.

PUBLISHER

TALKING ELECTRONICS is designed by Colin Mitchell of CPW INDUSTRIES, at 35 Rosewarne Ave., Cheltenham, Victoria, 3192. Australia. Articles suitable for publication should be sent to this address. You will receive full assistance with final presentation. All material is copyright however up to 30 photocopies is allowed for schools and clubs.

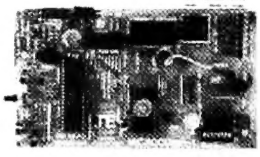
★Maximum recommended retail price only.

INDEX

Registered by Australia Post
Publication number VBP 4256

TECHNICAL *Ken Stone*
ARTWORK *Paul Loiacono*
ENQUIRIES *10 minute queries will be answered on* **584 2386** *8am - 6pm.*
ADVERTISING **(03) 584 2386**

P.59



We now have TWO micro-processor-based projects. One helps the other. Programs for the MICROCOMP-1 are produced on the TEC and run on the 'COMP. This will enable you to design all sorts of programs, which were hitherto impractical.

P. 9

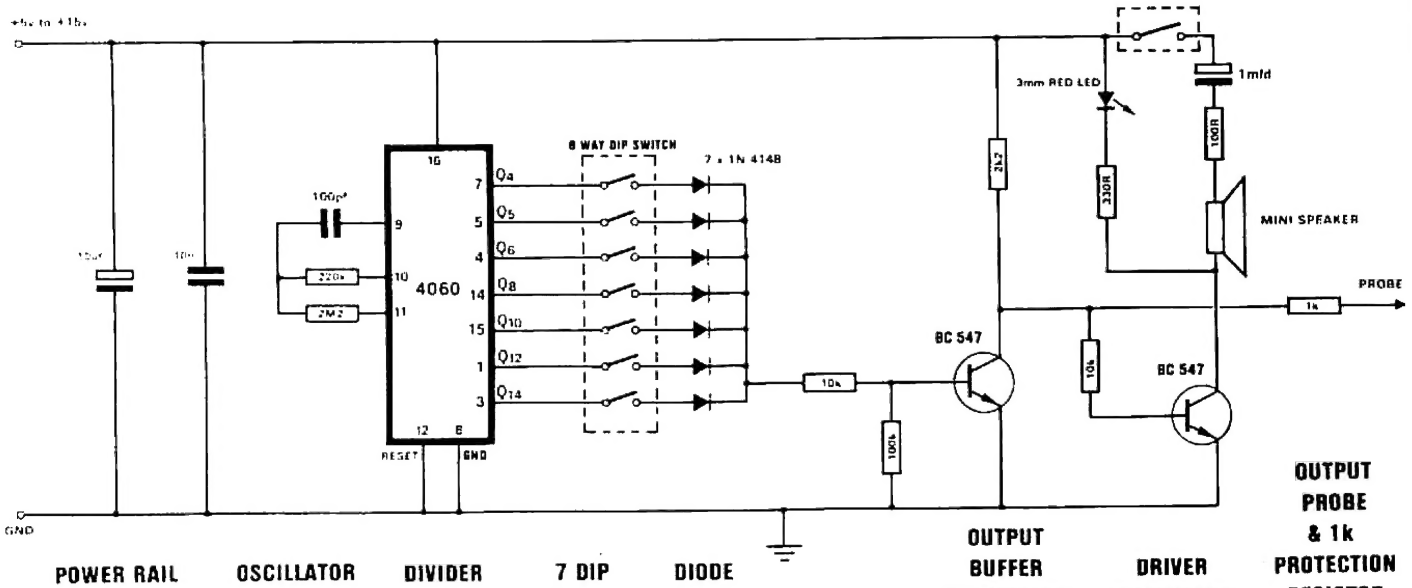
Printed Web Offset By:
Standard Newspapers Ltd.,
10 Park Rd, Cheltenham, 3192.

Distributed in Australia by Gordon & Gotch.

THE LOGIC PULSER

PC Board: $2.10
Kit of parts: $10.20

THE SECOND IN OUR 'TEST PROBE' TRIO . . .



LOGIC PULSER CIRCUIT

This is the second project in our test equipment trio.

It is a LOGIC PULSER capable of delivering pulses of various compositions, to any type of circuit you wish to test.

Basically it is designed to complement the LOGIC PROBE and can be used in situations where the LOGIC PROBE is not so effective.

In the next issue we will conclude the trio with a CONTINUITY TESTER - a test piece capable of indicating when a very low resistance is present.

It is an improvement over a multimeter in that it has an audible output and is NOT triggered when measuring across a diode.

Don't underestimate the importance of these items of test equipment. They are ALL needed to properly test and locate a fault in digital circuits.

The time when you appreciate them most is when a tricky fault comes along.

We have had a number of these and know how difficult it is to improvise.

A multimeter and CRO are all right for some applications but when it comes to testing digital circuits, they can give a readout which can be incorrectly interpreted and cause you to branch in the wrong direction.

The right tool for the right job is the answer and our trio is specifically designed for digital circuits.

PARTS LIST

1 - 100R
1 - 330R
1 - 1k
1 - 2k2
2 - 10k
1 - 100k
1 - 220k
1 - 2M2

1 - 100pf
1 - 10n greencap
1 - 1mfd electro
1 - 10mfd electro
7 - 1N 4148 diodes
1 - 3mm red LED
2 - BC 547 transistors
1 - CD 4060 IC
1 - mini speaker
1 - 8-way DIP switch
1 - 16 pin IC socket
5cm tinned copper wire
1 - paper clip for probe tip
1 - 50cm red hook-up flex
1 - 50cm black hook-up flex
1 - red EZY-CLIP
1 - black EZY-CLIP

1 - PULSER PC BOARD

The three we are presenting are available commercially for about $300, the set. We have been complemented on the logic probe by builders who prefer ours to commercially available units costing $200! Its audio output feature is very handy and only available on very few fully-assembled models.

The value of a logic PROBE is obvious. It picks up HIGHs and LOWs. But more important, it stretches short pulses so that they can be seen by the eye on an indicator LED.

This is the feature of the PULSE LED.

But the value of a LOGIC PULSER is not so obvious. You may think a probe is sufficient for all situations.

This is not so. Some circuits require to be tested in sections. This means the input waveform is not present and you require to know if the circuit will process the signals when it all goes together. This is the case for a logic pulser.

It is capable of delivering pulses to a circuit so that the result(s) can be detected, even though the circuit may not be complete.

The pulser has a wide range of output frequencies, ranging from 1Hz to 800Hz. Between these there are a number of output values which have varying mark-space ratios. The waveform, in all cases, has a fast rise and fall characteristic, making it ideal for injecting into chips.

The output transistor in the probe is protected via a 1k resistor so that you can place the tip on any pin of an IC and not damage either chip or probe.

HOW TO USE THE LOGIC PULSER

Connect the positive and negative leads of the pulser to the power rails of the project under test.

Turn the project ON and this will also supply power to the pulser.

Depending on the fault you are tracing, the frequency of the pulser (and the result you will get), will vary.

In fact there are so many different effects that we could not list them all. This is because the probe has a varying mark-space ratio which will create different effects in different circuits.

USING THE PULSER

The logic pulser can be used by itself if the project you are testing has output devices which show the results of the signals.

If output indicators are not available, you will have to use the logic probe described in issue 10.

The basic method of using the logic pulser is as follows:

Place the pulser on each of the pins of the chip and detect the effect on the output via the logic probe.

You may or may not get a reading. This will depend on how firmly the input lines are 'tied' to the rails.

This may sound unusual but it is an unfortunate fact for digital circuits. If you take the simple case of a chip being clocked by the output of another, the clock line is being taken HIGH and then LOW during the operation of the circuit.

If you halt the circuit when the output is LOW, it will be very difficult to pull the line HIGH because it is being kept LOW by the output transistors of the driving chip.

These transistors have a certain amount of ability to keep the line LOW and this is called SINKING ABILITY. For TTL this can be as much as 45mA and for CMOS it will be up to about 25mA.

This means we would have to put a signal on the line and deliver more than 25 mA or 45mA to pull it HIGH.

The output transistors would not like this and having a capability of delivering 45mA to a circuit could prove to be damaging to lots of other parts of the circuit.

The LOGIC PULSER is built around a CD 4060, 14-stage divider chip which has an inbuilt oscillator. With the addition of 3 components, the oscillator drives a string of 14 flip flops. The first output pin comes from the output of the 4th flip flop and this means the clock frequency is divided by 8.

From there it is divided further and an output is available from flips flops 5 to 14 (except 12).

We have used Q4, Q5, Q6, Q8, Q10, Q12, and Q14 in our project and have found these to be the most suitable for producing pulse trains.

Each output of the chip is taken to a switch on the 8-way DIP switch package and then diode gated together to form an OR gate.

A transistor buffer passes the signal to the probe tip where a 1k resistor is present to prevent damage to the probe if probing a power rail etc.

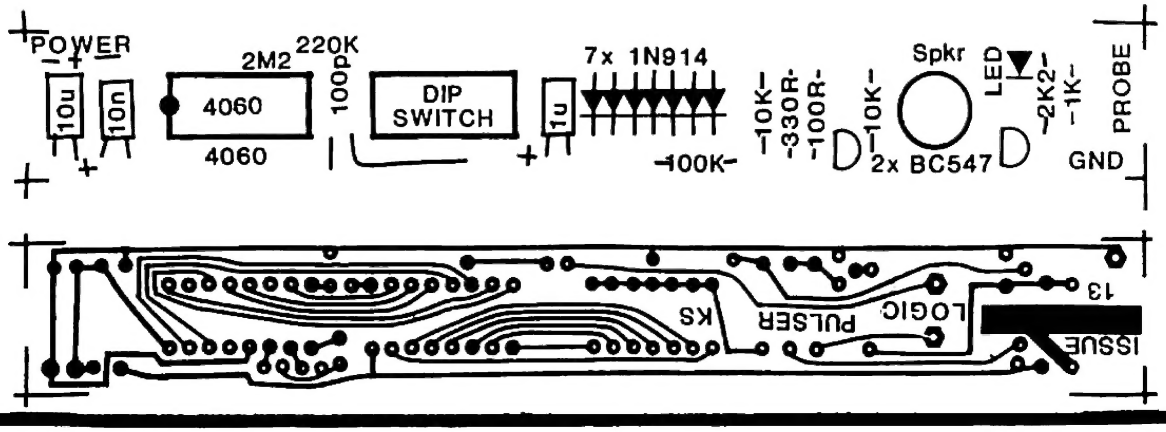
By turning ON various combinations of switches, you can produce a wide range of tones from a whistle to a 1Hz pulse.

You can modulate the tone by tuning on additional switches to produce 'chirping' or 'phone-ringing' tones. This has the effect of delivering a HIGH pulse modulated by LOW pulses, at the same time.

CONSTRUCTION

The PULSER is constructed on a single-sided PC board and assembled in the normal way.

Because the PC must be inserted into a toothbrush case, the components must be small and any tall components must be bent over to lay flat against the board.



There is nothing 'special' or 'unusual' about assembly and construction is straight-forward providing you have a fine tipped soldering iron.

The IC lands are circular due to the compact nature of the layout and you must be careful when inserting the IC socket.

All components mut be pushed hard against the PC board before soldering so that they take the least height possible. The two electrolytics, however, and 10n greencap must be kept slightly above the board so that they can be bent over and lay flat.

Start assembly by inserting the IC socket. If the socket has a cut-out indicating pin 1, place this over the 'dot' on the overlay.

Next fit the two links. They are positioned near the end of the IC socket. The second link is NOT straight but shaped like the letter 'L'. This has been necessary due to the compact nature of the layout.

Next fit the 7 gating diodes. These all face the same way and are pressed firmly against the board before soldering.

Between the diodes and the end of the board are 7 resistors (and a mini speaker). The resistors fit flat against the board and are soldered in place.

The two BC 546 or 547 or 548 transistors are pushed nearly up to the board and soldered in place.

The 3mm LED is the next to be placed on the board and it must be inserted the correct way for it to operate.

Insert the mini speaker and solder it in place. It can be fitted around either way.

Only a few components remain. Start with the DIP switch. It is soldered directly to the board so that the numbers on the switches can be read as per the photo.

Solder the two electrolytics so that the leads can be bent over and allow them to lay flat against the board. The 10mfd electrolytic fits near the end of the board and the 1mfd near the middle.

The 10n greencap also lays flat against the board. Make sure there is sufficient lead for this to be done.

Finally the two resistors and 100pf ceramic are fitted to the board. The resistors stand ON END to occupy the least space.

Nearly the end of construction. The power leads which supply the pulser with energy should be about 60cm long to give you plenty of freedom when using the unit. Solder alligator clips or E-Z clips to the ends. Before connecting the leads to the board, push them through two holes in the end of the case and then solder to the board.

Straighten out a paper clip and solder it to the end of the board to form a probe. Insert the 4060 IC and the project is ready for testing.

TESTING

Connect the probe to a power source. This can be a voltage from 5v to 15v. Determine which way the switches must be clicked for them to be ON so that you know how many switches are needed to achieve a particular pulse train.

Switch number 1 turns on the speaker, all others modulate the output.

Turn on switch '1' and any of the others. You will hear a tone in the speaker. If no tone is heard, try different combinations and observe the LED. If it blinks on and off, the fault will lie in the speaker circuit: made up of the speaker, 100R resistor, 1mfd electrolytic and switch 1. If the LED remains ON, the fault will lie in the chip and oscillator stage. If the LED remains OFF the fault may lie in the driver transistor.

To determine the operation of the 4060, connect the LOGIC PROBE, as described in issue 10, to the power rails and probe output pins 1, 2, 4, 5, 7, 14 and 15.

If no pulses are detected on these pins, the 4060 may not be oscillating. This could be due to the 3 frequency setting components not being wired into the circuit correcly, pin 12 not being connected to 0v or pin 16 not being connected to positive rail. Also look for shorts between pins 9, 10 and 11.

If a signal is present on the cathode end of the diode but not on the probe tip, the fault will lie in the buffer (amplifier) transistor. You cannot probe the base of the transistor with the Logic Probe as the voltage swing is not sufficient to be detected.

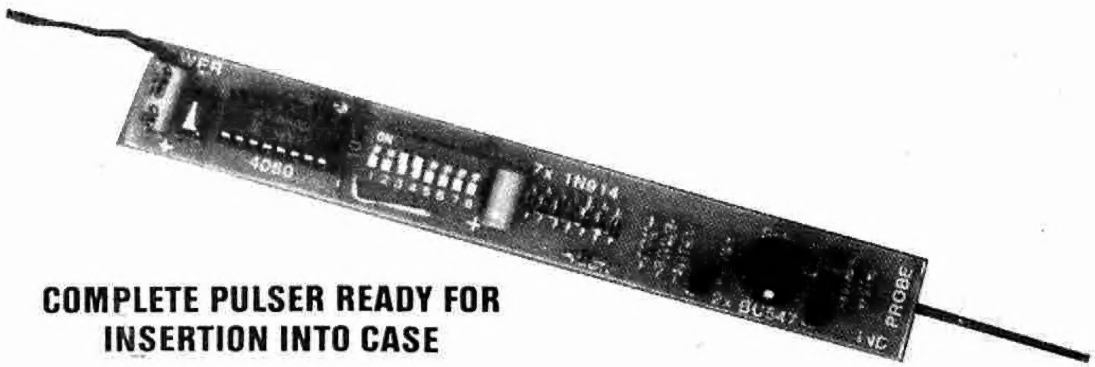
The transistor may be at fault or you may have a short between base and emitter. It could also be the 2k2 resistor may not be connected to the collector.

When there is no tone being produced by the pulser, the light emitting diode is illuminated, indicating the probe tip is HIGH.

The pulser can then be used as a HIGH to test various components, remembering it has a safety resistor in series with the tip to limit the current.

ASSEMBLY

The PULSER is designed to fit into a toothbrush case. You will have to look around your local chemist for an injection-moulded polystyrene case with the correct dimensions. Don't get a blow-moulded case or fabricated case as they don't have the rigidity.



COMPLETE PULSER READY FOR INSERTION INTO CASE

The spring-steel probe tip is passed through a small hole in the end of the case and the two power leads through two holes at the other end.

Close the case and you will find the switches cannot be adjusted.

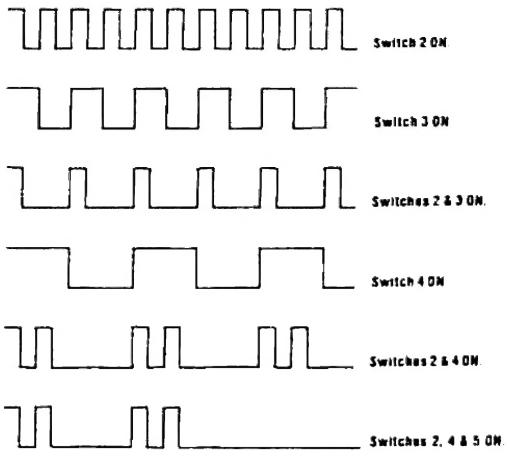
With a hot soldering iron, mark a rectangle above the switches, large enough to get the tip of your finger through. If you prefer, you can make the cut-out smaller and use a screwdriver to flick the switches.

Remove the board and fit the two halves of the case together. Use the tip of the soldering iron to cut through the plastic and create the cut-out. You can finish the edges with a file but make sure you don't fracture the case.

Insert the PC board and close the case again. Use clear sticky tape around the join and cut around the rectangular opening with a blade.

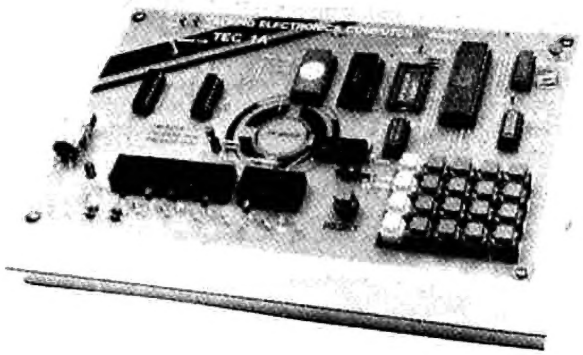
The probe is now ready to go to work on the next faulty project.

Let's hope that's soon!



These CRO diagrams show the effect of turning ON various switches. The correct combination is obtained by experimentation and you will get different effects on the output of the chip you are testing, as the switches are altered.

The most noticeable differences will occur when a capacitor is present in the circuit under test as it will will introduce a delay factor.



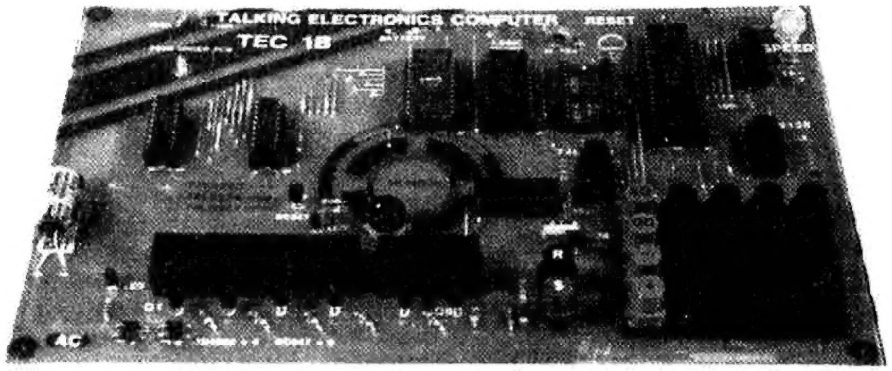
$98 COMPLETE
PLUS $6.00 POST

TALKING ELECTRONICS COMPUTER

PART IV

TEC-1A
&
TEC-1B

TEC 1A's can be converted to TEC 1B's by adding 1 push button, 1 47k resistor and 1 diode. Update to MON 2 and you have a SHIFT key for functions such as INSERT, DELETE etc.



TEC-1B board with SHIFT and RESET keys in foreground.

PC board: $21.00
Parts for 1B: $77.00
Case: $21.50
Post: $6.00 MAX.

FEATURES IN THIS ISSUE:
★ NON-VOLATILE RAM
★ EPROM BURNER

SEE ALSO:
TEC POWER SUPPLY on P. 23.

This is the fourth article on the TEC and introduces you to more Machine Code programming as well as two valuable add-ons.

The NON-VOLATILE RAM has been a real boon for assisting in program preparation for the MICROCOMP-1 project described in this issue.

Program can be written directly into RAM and by changing the switch, the contents will be retained for up to a year via the batteries mounted on the board.

This is the answer to all those requests from constructors wanting a battery backed-up system or tape-save facility. When the TEC is turned off, the contents of memeory will be saved and thus allow you to move the TEC from one location to another.

The RAM can also be used in place of an EPROM for the purpose of getting a system up and running. When you are satisfied with the design, the program can be transferred to EPROM.

This is where our second 'add-on' comes in. We have designed an EPROM BURNER to fit on the EXPANSION PORT socket.

With all the add-ons connected to the TEC, it was soon realized that the power required was more than could be supplied from a plug pack or 2155 transformer.

This led us to design a power supply exclusively for the TEC and at the same time include all the voltage values needed for the various projects.

So far we need 5v for the electronics, 12v for the relays and 26v for the EPROM BURNER.

The TEC POWER SUPPLY is capable of delivering these and can be expanded to about 1.4 amps at 5v by paralleling two 2155's.

Don't forget, the DC current capability of a 2155 is .7amps and NOT 1 amp and this has been covered in a previous article starting on page 5 of issue 11.

As you can see, one thing leads to another and we have sufficient add-ons to turn the TEC into a powerful programming tool.

The TEC itself has changed too. From the original TEC model, we improved the layout and upgraded the output latches to modern 20 pin types and

mounted the regulator under the board so that it would not be broken off.

We have now upgraded the TEC to model 1B and this has seen the inclusion of a shift key.

This shift feature allows the keyboard to have a second command for each key and opens up a world of possibilities.

Two functions which have been lacking on the TEC are INSERT and DELETE. With the addition of the shift key, you will be able to make corrections to your programs and close up gaps as well as create locations for new instructions.

Those who have already built the TEC can add a shift key in one of two ways. The lower RESET key can be converted into a SHIFT function by wiring a resistor and diode into circuit and connecting to the computer. The only problem with this is the upper RESET button. It will be difficult to access when the Video Display unit is mounted over the Z-80/EPROM area.

A better solution is to drill 4 holes near the lower RESET button and add the necessary components under the board.

The shift function is software controlled and you will need the updated MON 2 to get the shift key to work.

The MON 2 also includes a few other improvements. The most noticeable of these is the location of the STACK. You will remember the original position of the stack is very close to the top of the 6116.

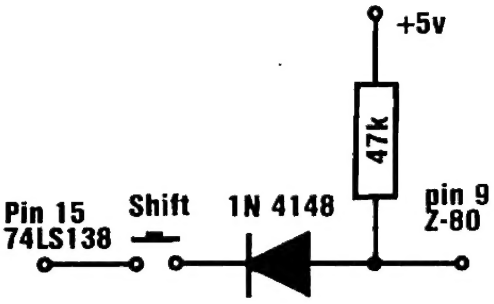
The main problem with this location is not knowing how far you can program before running the risk of hitting the stack.

The MON 2 places the stack at **08C0** and allows up to **C0** bytes to be stored. There is still a risk of crashing the computer if a stack error occurs as the stack grows down to **0000** and restarts at **FFFF** and will eventually hit the top of your program. Between C0 and **FF** is a storage area for pointers, restarts, display buffer, keyboard buffer, register save area and for interrupts.

This means programming starts at **0900** up to **0FFF** with the on-board 6116 and you don't have the problem of landing in the stack area.

You can upgrade to MON 2 by sending in your ROM and it will be re-burnt to MON 2. The cost is $3.50 plus $1.50 post.

A shift button, 47k resistor and signal diode is available in a separate kit for 80c and this will double the capability of your computer.



Adding Shift to TEC-1 , TEC-1A

MON 2 has 6 other shift functions and we are in the process of writing more software for further functions.

By the time this issue is released, we will have completed this writing and will include documentation with the chip.

The cost of the TEC has risen to $98.00 and it looks like going even higher as the exchange rate for the Aust. dollar drops. But we want to keep the computer below the magical $100 mark for as long as possible.

We have now supplied over 1,000 computers, in 3 different models. Only the earliest model has been fully documented. The upgraded versions vary only slightly and you should have no difficulty constructing them.

The reason for this is the simplicity of the board. Everything is fully identified on the overlay and requires only simple assembly.

Chances are it will operate first go but there is always a small possibility that something will be overlooked and it will not come on.

If you are caught in this situation, here is a run down on how to go about fixing it:

You will need a LOGIC PROBE and a MULTIMETER. A CONTINUITY TESTER (to be presented in next issue) will also be handy.

Firstly the visual checks:

If the displays fail to light-up and no sound is heard from the speaker, the most likely fault will be a broken track or poor solder connection. Turn the computer off and check each track with a multimeter switched to LOW-OHMs.

The regulator should get quite hot and should have 5v on the output lead. It must have at least 8v on the input lead to prevent voltage 'drop-out'.

The Z-80 will get quite warm, as will the output latch near the edge of the board.

The jumper near pin 1 of each latch should be checked. Only one must be inserted for each latch. This means you have two unused holes for each latch.

Check each of the keys for correct positioning. All flats must be DOWN.

The notch on each chip must also be DOWN.

Make sure all the pins of the IC sockets go through the holes in the PC board and are properly soldered. We have seen some pins doubled-up under the socket and not making contact with the tracks.

Check the capacitor near the speed control. It must be 100pf - not 100n. 100pf is indicated by '100' or '101' on a ceramic capacitor whereas 100n is shown as '104' on a mono block or 100nS on a blue body.

Check for non-soldered lands, missing links and incorrectly soldered links. We inspected one project in which the builder had cut the links to the exact length BEFORE soldering and consequently one link did not go through the board completely. It was too short to be soldered but the builder didn't notice. He soldered the land with the result that the link looked as though it was soldered!

Finally check for solder-bridges between adjacent lands with a multimeter set to LOW ohms. Remove the chips to get an accurate reading.

Now for the 'in-depth' diagnosis:

1. Turn the TEC on and check for 5v out of the regulator. Check POWER-ON LED. Check for 5v on each of the chips: 74LS 273 - pin 20. 2716 - Pin 24. 6116 - pin 24. Z-80 - pin 11. 4049 - pin 1 74LS138 - pin 16. 74LS923 - pin 20.

2. Check clock frequency by putting logic probe onto pin 6 of Z-80.

3. Check RESET pin of Z-80 is HIGH.

4. Check NMI line. (pin 17 of the Z-80). It will go LOW when a key is pressed. If not, a switch may be faulty or the keyboard scan oscillator may not be working. Keyboard oscillator is part of the 74C923 and the frequency-setting capacitor and debounce cap are the 100n and 1uf electrolytic.

5. Check pin 19 of the Z-80 with a logic probe. If it is not pulsing, program is not getting through.

6. Logic probe pin 18 of the 2716. Pulses on this pin show the ROM is being accessed.

7. Pulses on pin 18 of the 6116 show RAM is being accessed.

8. No pulses via checks 5, 6 or 7 indicate the full byte in an instruction is not getting through. This may be due to a faulty address or data line.

9. Check Do (pin 9 on the 2716, for continuity to pin 9 of the 6116 and also pin 14 of the Z-80.) Check the other 7 data lines for continuity and also the 11 address lines.

10. With all chips still in circuit, check each pin with the one adjacent to it, for the 2716, 6116 and Z-80. Our continuity checker in issue 14 will be ideal but if you can't wait, a multimeter can be used. Remember protection diodes are contained in most chips and low value resistors may be present on some lines. Low values of resistance may be perfectly acceptable - you are looking for zero ohms or short-circuits between tracks.

11. Check pin 20 of the Z-80 - the IN/OUT REQUEST line. If it is not pulsing, the output of the computer may be putting a load on the data bus.

12. Remove the two output latches and place the negative lead of a continuity tester on one of the pins. Touch every other pin of the output latch with the other lead. Move the first lead and repeat until all pins have been tested. Do the same with the other latch.

This will check for shorts on the data bus as well as between pins of the display.

13 If these fail to locate the fault, ring us at TE. We may be able to help you over the phone. If not, send the TEC in a jiffy padded bag and we will see what the trouble is.

So far we have had about 20 TECs sent in for checking and repair. About 8 of them suffered from voltage surges. This occured when the constructor shorted leads together and/or dropped a screwdriver on the back of the board when the TEC was operating. This can damage the EPROM, RAM and even the Z-80.

Don't let leads from the 'add-ons' dangle over the rest of the computer or let the SELECT leads touch each other when fitting them over the pins on the PC board.

The TEC is really very robust and we haven't damaged a unit yet, even though we have three in constant use and they are let running both day and night.

If you are careful with construction the TEC will work. But as with all pieces of electronic equipment, excess voltage will sound a death knoll.

While on this subject, we repaired two more unusual faults this month.

Both problems were the same and occured like this:

When the constructor was building the TEC, one or more of the components were soldered without being fully pushed onto the board.

Some time later the constructor discovered the fault and proceeded to push the component into place while trying to resolder the joint.

The result was the land broke away form the copper track and created a hairline fracture which was not spotted.

If this occurs on either the address or data bus, the TEC will fail to come on.

If this happens, the first pin to check is each of the Chip Enable pins on the two output latches.

If a probe on these pins show they remain HIGH, they are not being accessed.

Next check the IN/OUT select chip (below the expansion port) and see if it is being activated by the Z-80. No information on pin 4 could indicate that the program is not getting to the Z-80.

This leads you to suspect either one of the data lines or one or more of the address lines. They may be broken, with the result that the Z-80 is not receiving a full byte of program.

Before you jump to this conclusion, check the Chip Enable pin of the EPROM (pin 18) and see that it is LOW. This will mean the 2716 is being accessed and it should be talking to the Z-80.

If the Chip Enable pin is HIGH, go to the ROM/RAM decoder (below the clock chip) and check pin 4 to see that the pin is being accessed.

If one bus line is missing, the Z-80 will get the wrong op-codes and the program will not flow correctly.

Before we continue with programming, here are a few notes on assembling the TEC-1B as some changes have been made since the original notes in issue number 10.

The regulator is placed under the PC and bolted to the board via a 6BA nut and bolt. You can add heat fin if a number of add-ons are to be driven, but under normal circumstances, the regulator and board will dissipate the 1½ to 2 watts of heat.

The electrolytic has been changed to 1000mfd 25v and it lays flat on the board to keep a low profile.

The display drivers are slim-line types and 3 alternatives have been allowed for in the PC pattern. The overlay shows which links are to be added for the type chosen. Only ONE link must be used for each chip.

Finally a Z-80 or Z-80A can be used as the CPU chip. We are operating the TEC at 100kHz to 500kHz and this is well below the maximum speed for either type. A Z-80 will operate up to 2.5MHz and Z-80A up to 4MHz.

If any of the keys become worn, their contacts become erratic and sometimes a double-entry occurs. This can be overcome by increasing the value of the 1mfd on the 74c923 keyboard encoder to 4.7mfd or even 10mfd. This will mask out the contact bounce and produce a single pulse.

A 100n up to 10mfd can be used across the reset and it may be necessary to use the higher value if the Z-80 does not reset properly.

A 10k or 20k cermet can be used as the speed control and it can be either a VTP or HTP type. The advantage of a cermet means you can use your fingers to turn the pot and don't require a small screwdriver.

SHIFT

The latest addition to the TEC software is a SHIFT function.

This enables the number of functions to be increased from 4 to 24.

It means each of the buttons can be programmed to perform a second function when combined with the SHIFT button.

To access this second function the SHIFT button must be pressed first and kept pressed while the desired key is pressed.

HOW DOES IT WORK?

The keyboard encoder uses 5 lines of the data bus and the remaining 3 lines are not used.

The SHIFT button is connected to one of these lines and the monitor program re-written to detect its status when the keyboard is read.

Five functions are currently available. More are in the pipeline and their details will be explained in future articles.

The 5 functions are:

SHIFT +
This is the **INSERT** function. It moves every byte in the program up to the next higher location and inserts **00**

into the present address. This operation can be repeated any number of times to produce empty locations.

We have mentioned MON 2 allows programming to start at **0900** and the shift function operates in the area **0900** to **4000.** Addresses above **4000** are not catered for by the software but can be included if required.

Addresses below **0900** may cause a systems crash if you try to insert in this area as it is reserved for scratch pad, pointers and stack etc. Data below **0800** cannot be shifted as it is in ROM.

SHIFT – (shift, MINUS)
This is the **DELETE** key. It performs the opposite of INSERT. The data at the address currently being displayed is removed and all data above this address (and below **4000**) will be shifted DOWN one location. **3FFF** is loaded with **00.**

SHIFT ADdress
This function enables you to jump quickly to a particular location. Suppose you require to address **0A00** on a number of occassions. By pressing **SHIFT ADdress** the micro will jump to **0A00.** For this to happen, you must load a pointer location with the value **0A00,** then every time the SHIFT ADdress buttons are pressed, the display will show **0A00.** The pointer area is two bytes of memory located at **08D2 and 08D3.** By placing the **JUMP ADDRESS** at this location, the operation will be carried out.

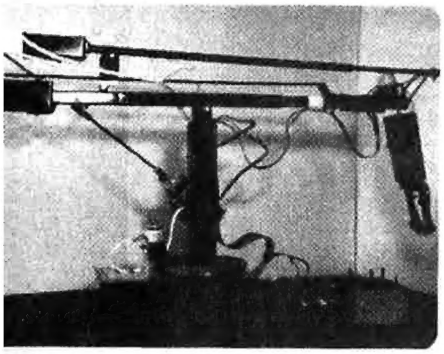
We are loading these two locations directly into BC register pair via a 4-byte instruction **ED 4B D2 08** and for the register pair to be correctly loaded, we must place the lower byte first in memory and then the high byte. This means we must load location **08D2** with **00** and **08D3** with **0A.**

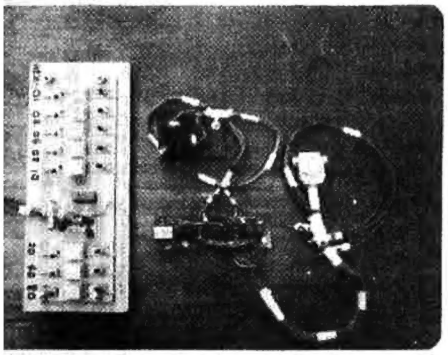
SHIFT 3
This function works exactly like SHIFT ADdress and enables you to have a second address to jump to. This time the pointer area is at **08D4** and **08D5.**

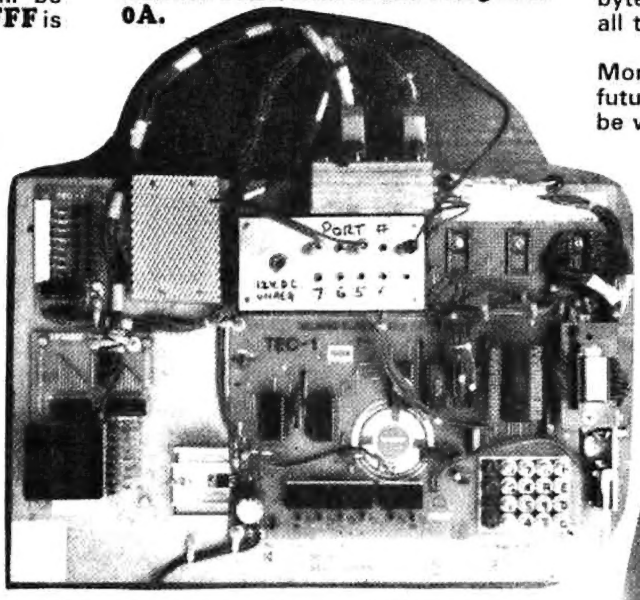
SHIFT 0
This is a search function. If you want to locate a value in a program or table, you could step through until it is located. This could take a long time. But with this function the value can be found very quickly. You can also locate the address of every other time it appears in a program.

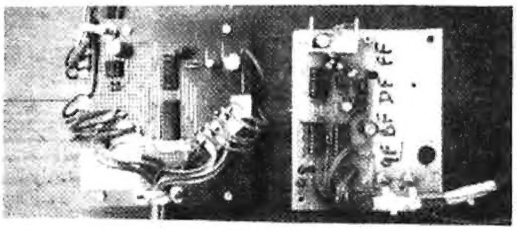
The value of the byte you are looking for is placed at **08E1.** Address the program you are testing and push **SHIFT 0.** The display will illuminate with the address of the byte you are looking for. Pushing SHIFT 0 again will display the second address of the byte. This can be continued to locate all the addresses.

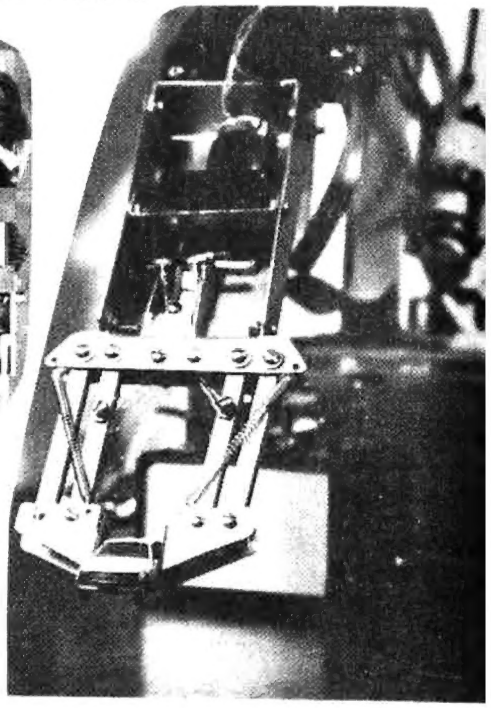
More function will be inclused in future monitors. Any suggestions will be welcome.

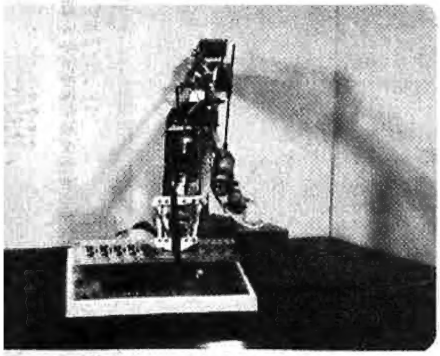


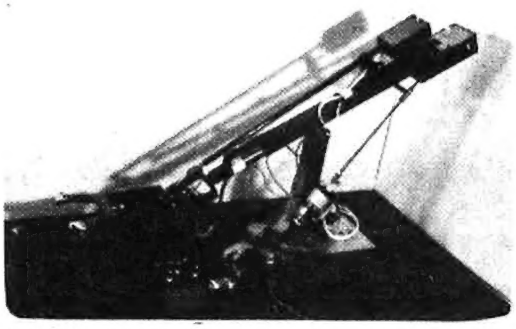


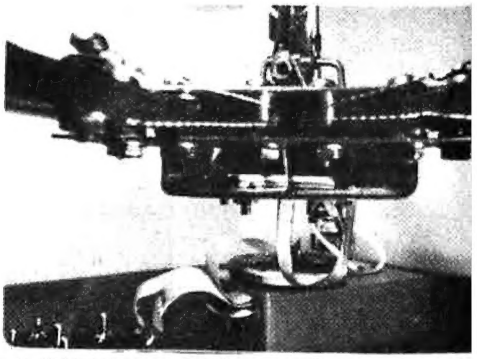












These photos show our science/electronics/ computer teacher's add-ons to the TEC and Glen Robinson's Robot Arm. It is made entirely from easy-to-obtain hardware parts, gears, motors and sturdy pieces of steel. A larger photo will do it more justice and this we will show in the next issue.

SIMPLIFYING PROGRAMS

One of the most important features of machine code is the fact that it occupies the least amount of memory.

The skill is to make use of this fact.

If we take the simple program from issue 12 page 21 (1st column), **RUNNING SEGMENT A ACROSS THE SCREEN**", we can shorten the program by using the following set of instructions:

LD A,01	**800**	**3E 01**
OUT (02),A	**802**	**D3 02**
OUT (01),A	**804**	**D3 02**
PUSH AF	**806**	**F5**
LD DE 20FF	**807**	**11 FF 20**
DEC DE	**80A**	**1B**
LD A,E	**80B**	**7B**
OR D	**80C**	**B2**
JR NZ 080A	**80D**	**20 FB**
POP AF	**80F**	**F1**
RLCA	**810**	**07**
JR 0804	**811**	**18 F1**

This program saves 8 bytes but has the disadvantage that the delay routine cannot be used by any other programs as it is hidden in the listing.

The delay could be placed apart if desired.

Eight bytes may not seem many to save but is a start to efficient programming.

This is where the byte-saving occured:

The instruction RLCA is a one-byte instruction to shift the contents of the accumulator left. (It does not shift through the carry bit but sets it, as explained in data sheet 13.)

The listing contains a number of **JR** instructions (and a displacement byte). These are 2 byte instructions whereas a **CALL** instruction requires 3 bytes.

THE DISPLACEMENT BYTE.

As listings get longer and more complex, the value of the displacement byte requires a method for determining its value.

When the jump is 5, 10 or 15 bytes forward or backward, the displacement value can be obtained by counting the locations: such as 00, 01, 02, 03 or FE, FD, FC, FB, FA etc. But when the jump is 20, 30 or more locations, the value can be obtained via a simple mathematical procedure.

Determining the value of the displacement byte requires 6 steps. By following these you cannot make a mistake.

Step 1. Count, via normal counting, the number of bytes between the displacement byte and the location being jumped to. Include the location you wish to land on. e.g: Take the following example:

11 FF 20
1B
7B
B2
20 dis

The number of bytes between **dis** and **1B** are: **20, B2, 7B, 1B.** These are counted as 1, 2, 3, 4. Thus the answer is 4.

We will select a higher value for our problem to emphasise the need for the procedure.

Suppose the number of locations we wish to jump back is 49.

Step 2: Convert 49 to a HEX value by dividing it by 16:

The answer is 31H

Step 3: Convert 31H to binary:

3 1
0011 0001

Step 4: Change each 1 to 0 and each 0 to 1:

Ans: 1100 1110

Step 5: Add 1 to the answer:

Ans: 1100 1111

Step 6: Convert to a HEX value:

C F

This is the value of the displacement byte required to achieve a backward jump of 49 bytes.

The machine code instruction will depend on the **JR** condition and will be one of the following:

28 CF, **20 CF,** or **18 CF**

The steps we have performed are called **TWO s COMPLEMENT.**

Using the knowledge we have gained, we will improve the **BACK and FORTH** program from P 15 of issue 12.

Mainly aiming at byte reduction, we will include a **BIT TESTING** instruction to prevent overshoot of the displays. Bit 0 in the accumulator is tested and if it is a '1', the program will cause a change in direction by rotating the accumulator in the opposite direction.

With these alterations in the program we will save about 12 bytes. Try the program:

LD A,01	**800**	**3E 01**
OUT (02),A	**802**	**D3 02**
OUT (01),A	**804**	**D3 01**
CALL DELAY	**806**	**CD 00 0A**
RLCA	**808**	**07**
BIT 6,A	**809**	**CB 77**
JR Z 0804	**80B**	**28 F6**
RRCA	**80D**	**0F**
OUT (01),A	**80E**	**D3 01**
CALL DELAY	**810**	**CD 00 0A**
BIT 0,A	**813**	**CB 47**
JR Z 080D	**815**	**28 F6**
JR 0809	**817**	**18 F0**

at 0A00:

F5
11 FF 20
1B
7B
B2
20 FB
F1

The program is required to test bit 6 in the accumulator. If it is found to be a '1', the contents of the accumulator is shifted in the opposite direction. Bit 0 is then tested and when found to be '1', the program jumps back and shifts the accumulator in the original direction.

BYTE TABLE. To use this table, the byte following the JR instruction is counted as BYTE ZERO. From this byte you count in either the positive or negative direction using decimal counting.

0	00	48	30	96	60	1	FF	-49	CF	97	9F
1	01	49	31	97	61	-2	FE	-50	CF	98	9E
2	02	50	32	98	62	3	FD	-51	CD	99	9D
3	03	51	33	99	63	4	FC	-52	CC	100	9C
4	04	52	34	100	64	-5	FB	-53	CB	101	9B
5	05	53	35	101	65	6	FA	-54	CA	102	9A
6	06	54	36	102	66	7	F9	-55	C9	103	99
7	07	55	37	103	67	8	F8	-56	C8	104	98
8	08	56	38	104	68	-9	F7	-57	C7	105	97
9	09	57	39	105	69	10	F6	-58	C6	106	96
10	0A	58	3A	106	6A	11	F5	-59	C5	107	95
11	0B	59	3B	107	6B	-12	F4	-60	C4	108	94
12	0C	60	3C	108	6C	-13	F3	-61	C3	109	93
13	0D	61	3D	109	6D	-14	F2	-62	C2	110	92
14	0E	62	3E	110	6E	15	F1	-63	C1	111	91
15	0F	63	3F	111	6F	16	F0	-64	C0	112	90
16	10	64	40	112	70	17	EF	-65	BF	113	8F
17	11	65	41	113	71	18	EE	-66	BE	114	8E
18	12	66	42	114	72	19	ED	-67	BD	115	8D
19	13	67	43	115	73	20	EC	-68	BC	116	8C
20	14	68	44	116	74	21	EB	-69	BB	117	8B
21	15	69	45	117	75	22	EA	-70	BA	118	8A
22	16	70	46	118	76	23	E9	-71	B9	119	89
23	17	71	47	119	77	24	E8	-72	B8	120	88
24	18	72	48	120	78	25	E7	-73	B7	121	87
25	19	73	49	121	79	26	E6	-74	B6	122	86
26	1A	74	4A	122	7A	27	E5	-75	B5	123	85
27	1B	75	4B	123	7B	28	E4	-76	B4	124	84
28	1C	76	4C	124	7C	29	E3	-77	B3	125	83
29	1D	77	4D	125	7D	30	E2	-78	B2	126	82
30	1E	78	4E	126	7E	31	E1	-79	B1	127	81
31	1F	79	4F	127	7F	32	E0	-80	B0	128	80
32	20	80	50			33	DF	-81	AF		
33	21	81	51			34	DE	-82	AE		
34	22	82	52			35	DD	-83	AD		
35	23	83	53			36	DC	-84	AC		
36	24	84	54			37	DB	-85	AB		
37	25	85	55			38	DA	-86	AA		
38	26	86	56			39	D9	-87	A9		
39	27	87	57			40	D8	-88	A8		
40	28	88	58			41	D7	-89	A7		
41	29	89	59			42	D6	-90	A6		
42	2A	90	5A			43	D5	-91	A5		
43	2B	91	5B			44	D4	-92	A4		
44	2C	92	5C			45	D3	-93	A3		
45	2D	93	5D			46	D2	-94	A2		
46	2E	94	5E			47	D1	-95	A1		
47	2F	95	5F			48	D0	-96	A0		

INTRODUCTION TO COUNTING

A microprocessor system is ideally suited to counting situations. It can be programmed to count to any particular number then sound an alarm or operate a relay or even notify the near-completion of a run.

It can count UP or DOWN as well as count in sub-multiples.

Take the case of packing a box of TE magazines.

Firstly the operator requires a count of 10. Each 10 issues must be placed in opposite directions in a box to produce a level stack. The operator then needs to know when a count of 140 is reached, which represents a full box.

Finally the packers need to know how many boxes of magazines have been packed so that the delivery docket can be filled out.

This is effectively 3 counters which must be interconnected to achived the required result. Ideally an audible signal should be produced at the end of each count of 140 so that the packer(s) can concentrate (day dream) on the job.

The chance of finding such a design is almost nil, except via individual modules which will have to be connected together to create the system. The cost of doing this would be about $300!!

But with a microprocessor system such as the TEC, all these up-down requirements are possible in the one unit, by simply providing a program!

The art of producing a suitable program is the content of this section.

We will start from the beginning and explain how counting is achieved, how to interface a 'count-button' and progress to producing a 3-digit up-down counter.

A count-down system is often used as it can be pre-programmed with a START VALUE and the counter decrements to zero. It then sounds a bell, activates a relay and resets to the pre-determined start-value.

After studying the 3-digit counter you will be able to create a 4, 5 or 6 digit counter and even incorporate sub-values to facilitate packing etc.

The counter can also be designed to have 2 concurrent tallies, one being permanently displayed while the other is available on call-up via the press of a button.

They would be displayed for a few seconds and fall back into memory.

Absolutely any combination, application or requirement can be catered for, it only requires programming.

To make it easy to understand, we have started with a simple program. But, as explained, this type of program soon runs out of capability. Thus a more complex system of time-sharing of the displays must be used.

But this too has limitations and finally an even more complex (as far as understanding is concerned) use of registers, must be employed.

With this high-level system, the scope is enormous. The system can be increased to 8 digits, two or more separate readouts, and have tally values available on call-up.

This is where we start . . .

Creating your own COUNTING MACHINE is one of the capabilities of our micro. You can produce a display which increments or decrements by a count of one or more on each press of a button. And the button doesn't have to be the '1' button. In our case we have used the '4' button to show that any button can be used.

By changing the values in the 'look-up' table, you can create the up or down condition - something which is virtually impossible with discrete counting-chip construction.

You can even produce letters of the alphabet and increment each time 'Z' or 'F' or 'X' appears. You can do anything from counting by 2's to dividing by 'Z'.

For our first exercise we will produce a counter which counts to 9. This is a very simple program. Only one display will be accessed and thus we can output to it so that it turns on HARD, while the computer is in the HALT mode, waiting for an interrupt from the keyboard.

It is important to note the computer does not produce the numbers 0-9, the program creates them. The table at **0900** contains values which turn on various segments of the display to create the numbers.

0-9 COUNTER

LD A,01	800	3E 01	The accumulator is loaded with **01** and outputted to port **01**. This connects the cathode of the first display to earth.
OUT (1),A	802	D3 01	
LD HL,0900	804	21 00 09	Load HL pair with the address of the number table.
LD A,(HL)	807	7E	Load the first byte of the number table into the accumulator.
OUT (2),A	808	D3 02	Connect segments of the display to the positive rail to get first number.
LD B,0A	80A	06 0A	Register B is our 'counting register'. It counts 10 bytes from **0900** to **0909**.
HALT	80C	76	HALT the program so that first number (0) will appear on the display.
CP 04	80D	FE 04	The program recognises only button '4'
JR NZ Halt	80F	20 FB	If not button '4', go to HALT. If button '4' pressed, increment HL to look at 0901.
INC HL	811	23	The byte at 0901 is loaded into the accumulator.
LD A,(HL)	812	7E	The value at 0901 (**28**) creates the figure '2' on the display
OUT (2),A	813	D3 02	Output **28** to port 02.
DJNZ Halt	815	10 F5	Register B is decremented and if it is not zero, the program goes to HALT.
JP Z 0800	817	CA 00 08	When register B is zero, the program jumps to START (**0800**).

at 0900

EB	=	0
28	=	1
CD	=	2
AD	=	3
2E	=	4
A7	=	5
E7	=	6
29	=	7
EF	=	8
AF	=	9

Type the program into the TEC and press RESET, GO. The number '0' will appear on the display.

Press various buttons on the key-board and notice that only button '4' advances the count.

Step through the table by pressing button 4.

1. Experiment with the program by creating the numbers on another display.
2. Create a down-count by inserting the table at **0900** in the opposite direction. i.e: **AF, EF, 29, E7, A7, 2E, AD, CD, 28, EB.**
3. Create a count-to-six by changing the value of **B** (**080A**) to **06.**
4. Create the letters A-F by adding their appropriate hex values to the table, select the correct value for **B**, change the compare value to enable button 'C' to operate and step through the table you have produced.

TWO DIGITS

When two or more digits are to be displayed, the program must contain a multiplexing or time-sharing arrangement so that each display can show a number from 0 to 9 without interfering with the other. This means a HALT instruction cannot be used as only one display will remain alight!

The program must be constantly looping or 'running' so that both displays are kept on. Each time the program cycles, it is looking for an interrupt from the keyboard and if one comes along, the program operates on the data it receives and compares it with the value 04. Depending on the result, the program will branch to one of two places.

The program below produces a count-to-99 using the '4' button as the input.

The basic structure of the program is quite simple and uses register pair HL to point to the address (at 0900) for the hex value needed to produce the numbers 0 to 9.

Register pair DE points to the hex value (again at 0900) needed to produce the 10's value.

Each of these register pairs are incremented and compared with FF to see if the end of the table has been reached. The increment of the DE register takes place when FF is detected on the 1's count. When the 10's count reaches the end of the table, the whole program is reset.

The computer does not know it is counting to 10. It merely knows it is incrementing through a table. You could put Chinese values on the display and count to 11, simply by changing the value of a few locations.

Here is the 0-99 program and an explanation of each step:

0-99 COUNTER

XOR A	800	AF	Set the accumulator to ZERO.
LD I,A	801	ED 47	Load the interrupt register with ZERO.
LD DE,0900	803	11 00 09	Load DE pair with address **0900.**
LD HL,0900	806	21 00 09	Load HL pair with address **0900.** END OF START-UP.
XOR A	809	AF	Beginning of MAIN PROGRAM. Clear Accumulator.
OUT (1),A	80A	D3 01	Turn OFF 1's display.
LD A,(HL)	80C	7E	Load accumulator with byte pointed to by HL pair.
OUT (2),A	80D	D3 02	Output to port 2.
LD A,01	80F	3E 01	Load accumulator with 1.
OUT (01),A	811	D3 01	Output accumulator to port 1. Display is illuminated.
LD B,10	813	06 10	Register B is a COUNT REGISTER. Load it with 10 to create 16
DJNZ FE	815	10 FE	loops to turn on 1's display.
XOR A	817	AF	Clear Accumulator.
OUT (1),A	818	D3 01	Output 0 to port 1 to turn OFF display.
LD A,(DE)	81A	1A	Load accumualtor with byte at 0900 etc as pointed to by DE pair.
OUT (02),A	81B	D3 02	Output the value thus obtained to port 2.
LD A,02	81D	3E 02	Load the accumulator with 2
OUT (01),A	81F	D3 01	Output to port 1 to turn on 10's display.
LD B,10	821	06 10	Load count register with 10 (decimal 16) and create 16 loops to
DJNZ FE	823	10 FE	turn on 10's display.
LD A,I	825	ED 57	Load the interrupt register into the accumulator.
CP 04	827	FE 04	Compare with 4 i.e. subtract 4 from I. If the result is ZERO,
JP NZ 0809	829	C2 09 08	advance to **082C** If the answer is NOT ZERO, go to **0809.**
XOR A	82C	AF	Clear Accumulator.
LD I,A	82D	ED 47	Load the Interrupt register with ZERO.
INC HL	82F	23	Increment register HL to point to address **0901** etc
LD A,(HL)	830	7E	Load the value at **0901** into the accumulator.
CP FF	831	FE FF	Compare the value obtained (eg **28**) with **FF.** If equal, advance to
JP NZ 0809	833	c2 09 08	**0836,** if NOT equal, go to **0809.**
INC DE	836	13	Increment DE.
LD A,(DE)	837	1A	Load the value pointed to by register DE into the accumulator.
CP FF	838	FE FF	Compare with **FF** to see if end of table has been reached
JP NZ 0806	83A	C2 06 08	If **FF** is reached, result will be zero. Advance to **083D. If not, go**
JP 0800	83D	C3 00 08	**JUMP TO START** **to 0806.**

at 0900:

EB = 0
28 = 1
CD = 2
AD = 3
2E = 4
A7 = 5
E7 = 6
29 = 7
EF = 8
AF = 9
FF

The **CONDITIONAL JUMP** instruction requires explanation.

In the 00-99 counter program above, there are three places where the Z-80 will jump to another part of the program when a certain condition is met. The condition is **NZ** (NON ZERO). Let us explain how to interpret this:

From the program above:

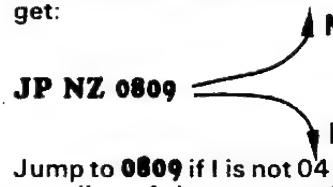
```
LD A,I
CP 04
JP NZ 0809
```

These 3 lines state: The I register is loaded into the accumulator. The accumulator is compared with 04. Jump to 0809 is the result is NON ZERO.

How does the COMPARE statement work?

The **CP** operation is carried out like a subtract operation and the zero flag (Z flag) will be SET if the result is ZERO and RESET if the result is NON ZERO. This means it will be '1' if the answer is zero and '0' if the answer is not zero.

This is quite confusing because you have to deal with the negative of a negative. To simplify things we can use the word **MET** for ZERO. Thus we get:



Jump to **0809** if I is not 04 or go to the next line of the program if I = 04.

Now we come to the THREE DIGIT COUNTER. It has an UP/DOWN facility as well as CLEAR. Push + for increment, − for decrement and push ADdress to zero the display. The counter can also be preset by loading **0B03** and **0B04** with values as shown in the listing on the right:

```
PUSH AF       A00  F5
CALL 0A0D     A01  CD 0D 0A
POP AF        A04  F1
RRA           A05  1F
RRA           A06  1F
RRA           A07  1F
RRA           A08  1F
CALL 0A0D     A09  CD 0D 0A
RET           A0C  C9
```

```
AND 0F        A0D  E6 0F
LD HL         A0F  21 00 0C
ADD A,C       A12  85
LD L,A        A13  6F
LD A(HL)      A14  7E
LD (BC)A      A15  02
INC BC        A16  03
RET           A17  C9
```

THREE DIGIT COUNTER

```
START  LD BC 0B00      800   01 00 0B
       LD DE 0B03      803   11 03 0B
       LD A(DE)        806   1A
       CALL 0A00       807   CD 00 0A
       INC DE          80A   13
       LD A(DE)        80B   1A
       CALL 0A0D       80C   CD 0D 0A
       LD HL 0B02      80F   21 02 0B
       CALL SCAN       812   CD 00 09
       LD A,I          815   ED 57
       LD HL 0B03      817   21 03 0B
INC    CP 10           81A   FE 10
       JRNZ DEC        81C   20 0D
       LD A(HL)        81E   7E
       INC A           81F   3C
       DAA             820   27
       LD (HL)A        821   77
       JRNC START      822   30 20
       INC HL          824   23
       LD A(HL)        825   7E
       INC A           826   3C
       DAA             827   27
       LD (HL)A        828   77
       JR CLEAR        829   18 19
DEC    CP 11           82B   FE 11
       JRNZ RESET      82D   20 0D
       LD A(HL)        82F   7E
       DEC A           830   3D
       DAA             831   27
       LD (HL)A        832   77
       JRNC CLEAR      833   30 0F
       INC HL          835   23
       LD A(HL)        836   7E
       DEC A           837   3D
       DAA             838   27
       LD (HL)A        839   77
       JR CLEAR        83A   18 08
RESET  CP 13           83C   FE 13
       JRNZ CLEAR      83E   20 04
       XOR A           840   AF
       LD (HL)A        841   77
       INC HL          842   23
       LD (HL)A        843   77
CLEAR  LD A,FF         844   3E FF
       LD I,A          846   ED 47
       JR START        848   18 B6
```

SCAN

```
LD B,04       900  06 04
LD A(HL)      902  7E
OUT (02)A     903  D3 02
LD A,B        905  78
OUT (01)A     906  D3 01
LD B 50       908  06 50
DJNZ          90A  10 FE
DEC HL        90C  2B
LD B,A        90D  47
XOR A         90E  AF
OUT (01)A     90F  D3 01
RRC B         911  CB 08
JRNC          913  30 ED
RET           915  C9
```

at 0C00:

```
EB
28
CD
AD
2E
A7
E7
29
EF
AF
```

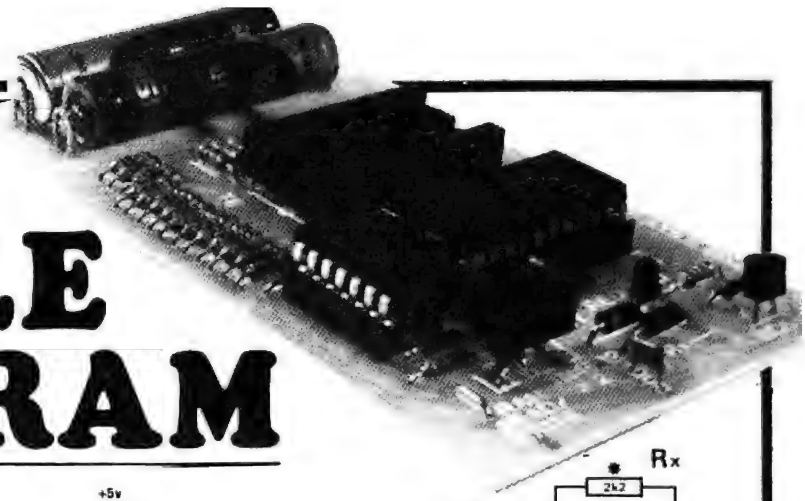
To make this program easy to understand, we have listed ONE COMPLETE CYCLE. Exactly as it is run by the computer. CALL ROUTINES have been included each time they are called and this makes the listing fairly long.

When the program is run for the first time, the display will show the values contained at **0B03** and **0B04**. For the purpose of showing how the program works, we will place **21** at **0B03** and **43** at **0B04**. This will cause the display to show 123 (the value 4 will not appear in this 3 digit counter).

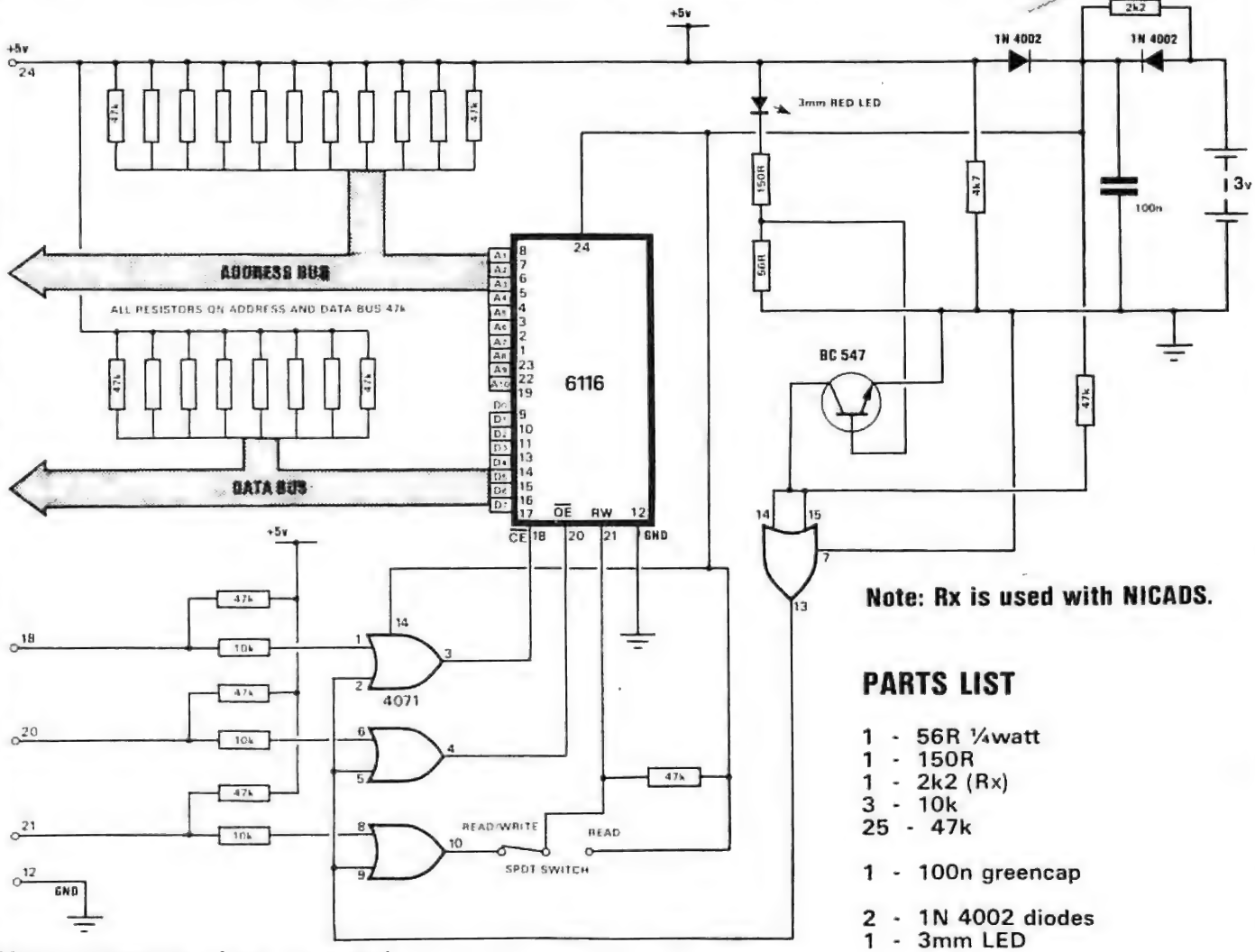
Follow through each of the steps and you will see how the program picks up data from the 'BUFFER ZONE' and converts it values which can be identified as numbers on the display. This program is being executed at more than 100 times per second!

```
START  LD BC 0B00      Location 0B00 stores the value of the units display
       LD DE 0B03      D is loaded with 0B and E is loaded with 03.
       LD A(DE)        Load two nibbles (21 in our example) into the accumulator
       PUSH AF         Save the accumulator
       AND 0F          This instruction zero s the high nibble leaving 01
       LD HL 0C00      H is loaded with 0C and L with 00
       ADD A,L         Add 00 to the accumulator to get 01 (01 is from above)
       LD L,A          Load the accumulator (it has 01 in it) into the L register
       LD A(HL)        Load the value at 0C01 (28) into the accum (HL is now 0C01)
       LD (BC)A        Load the value from the accumulator (28) into the BC register pair
       INC BC          Increment the BC register (it will become 0B01).
       POP AF          Fetch the accumulator (value 21) from the stack
       RRA             Shift the value 21 four places to the right
       RRA             so that the high bits will be transposed
       RRA             with the low bits. The result will be 12
       RRA
       AND 0F          Remove the 4 HIGH bits to get 02
       LD HL 0C00      H will be loaded with 0C and L with 00
       ADD A,L         Add 00 to the accumulator to get 02
       LD L,A          Load 02 into the L register
       LD A(HL)        Load the value at 0C02 (CD) into the accumulator
       LD (BC)A        Load the accumuloatr (it has 02 in it) into the address pointed to by BC.
       INC BC          Increment the BC register (to 0B02)
       INC DE          DE is incremented to 0B04
       LD A(DE)        The value at 0B04 (43) is loaded into the accumulator
       AND 0F          The HIGH nibble is cleared to get 03
       LD HL 0C00      H is loaded with 0C and L with 00
       ADD A,L         00 is loaded into the accumulator to get 03
       LD L,A          03 is loaded into L
       LD A(HL)        The value at 0C03 (AD) is loaded into the accumulator
       LD (BC)A        Load AD into location 0B02.
       INC BC          The BC register pair is incremented to 0B03
       LD HL 0B01      Load H with 0B and L with 02
       LD B,04         Load B with 04
       LD A(HL)        Load the accumulator with the value at 0B02 (AD)
       OUT (02),A      Output AD to port 02
       LD A,B          Load the accumulator with 04
       OUT (01),A      Output 04 to port 01. This will turn on a b c d g to get '3'
       LD B,50         B is loaded with 50hex (five-oh or 80 in decimal)
       DJNZ            Perform a jump command for 80 loops
       DEC HL          HL now points to 0B01
       LD B,A          The accumulator (it contains 04) is loaded into B
       XOR A           Clear the accumulator
       OUT (01),A      Turn OFF the display
       RRC B           Shift register B right to get 02 (half its previous value)
       LD A(HL)        Load the value at 0B01 (CD) into the accumulator
       OUT (02),A      Output the value CD to port 2
       LD A,B          Load B (02) into the accumulator
       OUT (01),A      Output 02 to port 1. This turns on the second display and a b d e g '2'
       LD B,50         Load B with 50 (in hex)
       DJNZ            Perform 50 loops. This is 80 loops
       DEC HL          HL now points to 0B00
       LD B,A          Load 02 into B
       XOR A           Zero the accumulator
       OUT (01),A      Turn OFF the display
       RRC B           Rotate register B to the right to get 01
       LD A(HL)        Load the value at 0B00 (28) into the accumulator
       OUT (02),A      Output 28 to port 2
       LD A,B          Load 01 into the accumulator
       OUT (01),A      Output 01 to port 1
       LD B,50         Load B with 50
       DJNZ            This instruction creates 80 loops of delay-time
       DEC HL          HL is decremented but the 4th loaction is not used as you will see
       LD B,A          Load 01 into B
       XOR A           Zero the accumulator
       OUT (01),A      Turn off the display
       RRC B           Register B is shifted and the carry bit is SET
       LD A,I          The accumulator is loaded with a value from the keyboard
       LD HL 0B03      H is loaded with 0B and L with 03
INC    CP 10           The value 10 is compared with the accumulator
       DJNZ            If the two are the SAME, the program increments. If not it jumps to DEC
       LD A(HL)        Load A with 21
       INC A           Increase the value 21 to 22
       DAA             Decimal adjust the accumulator if needed (not in this case)
       LD (HL),A       Load 22 into the location 0B03
       JRNC start      Jump to start is no carry from DAA operation. If a carry is produced, i.e.
       INC HL          when 99 advances to 100, increment HL to 0B11
       LD A(HL)        Load the value at 0B11 into the accumulator
       INC A           Increment A
       DAA             Decimal adjust the accumulator if necessary
       LD (HL)A        Load the accumulator into 0B04
       JR CLEAR        Jump to CLEAR
CLEAR  LD A,FF         Load FF into the accumulator
       LD I,A          Load the accumulator into the interrupt vector register
       JR START        Jump to START
```

NON-VOLATILE RAM



PC Board: $3.60
Parts: $19.60



Many constructors have requested some means of saving the programs they produce on their TEC'c.

Most suggested a TAPE SAVE facility whereby they could load their program onto a cassette and hold it until required. This would allow the TEC to be turned off / or used for other tasks.

Tape save is a project which will appear as a future add-on and has certain advantages. Before we present a tape save we have designed a storage project using a 6116 CMOS RAM chip, which will be the next best thing.

It is a battery backed-up RAM which can be written into and then protected via a switch to become a Read Only Memory. It can be left connected to the TEC or removed at any time and the battery automatically takes over, keeping the contents in an unchanged state.

The 6116 RAM takes an amazing 2 microamps in the storage state (power-down state) and this represents little more than the natural deterioration of the cells.

Under these conditions the two back-up cells should last about 1 year.

Note: Rx is used with NICADS.

PARTS LIST

1 - 56R ¼watt
1 - 150R
1 - 2k2 (Rx)
3 - 10k
25 - 47k

1 - 100n greencap

2 - 1N 4002 diodes
1 - 3mm LED

1 - BC 547 transistor
1 - CD 4071 IC
1 - 6116 RAM

1 - 14 pin IC socket
1 - 24 IC socket
1 - 24 pin wire-wrap socket
1 - 24 pin DIP header

1 - SPDT switch

2 - AAA cells

20cm tinned copper wire

1 - NON-VOLATILE RAM PC BOARD

The main advantage of this form of memory is information is immediately accessible and does not have any loading delays as experienced with tape.

When producing Machine Code programs, it is not necessary to have a large RAM memory and a single 6116 will be sufficient for even quite a long program.

We have called this project NON-VOLATILE RAM and have already found it to be invaluable when developing programs for other computers and dedicated systems. It is easy to use and can be written into directly or filled from TEC memory.

Once the data has been deposited, the switch is changed to 'ROM' position and the information is protected.

HOW IS THIS DONE?

The 6116 RAM chip is the centre of the design. It is a low-power CMOS device with exceptionally low stand-by current. Many of the TEC owners will already have one of these chips. It is important to note that only the 6116 can be used as the N-MOS version 58725 consumes 4,000 times more current in stand-by mode.

The 6116 draws about 2 micro-amps whereas the 58725 consumes 8 milli-amps. If you have one of each, use the 6116 for the non-volatile project and retain the 58725 as the TEC RAM.

Theoretically the RAM in the TEC can be converted to battery back-up but this poses a problem as the control lines must be taken HIGH or LOW to prevent the RAM being written over during the time when the TEC is powering down.

We had difficulty in achieving a guaranteed result and opted for a separate RAM card. This allows the card to be transferred to other projects, enabling programs to be generated and corrected until they operate perfectly.

When the RAM card is plugged into the TEC it draws power via diode D1 while diode D2 prevents the voltage from charging the batteries. When the TEC is switched off, the batteries supply a potential to the chip via diode D2. Diode D1 prevents the TEC from drawing on the batteries.

The 2.4volts from the batteries (.6v is lost across diode D2) is sufficient to hold the data.

The LED and resistors R1, R2 form a voltage-loss detection circuit to switch the non-volatile RAM into stand-by mode.

They form a voltage-divider circuit for the base of the BC 547 transistor. When the voltage across the LED and resistor R1 is below 4v, the transistor switches OFF, allowing the inputs of the OR gate to go HIGH, via a buffer. This takes the Chip Enable, Output Enable and Read/Write lines HIGH, protecting the RAM contents from erasure.

All address and data lines are taken LOW to prevent them floating and thus wasting power.

USING THE RAM CARD

The 'on-board' cells will provide power to the chip for about 1 year and this makes it an ideal storage medium for saving programs.

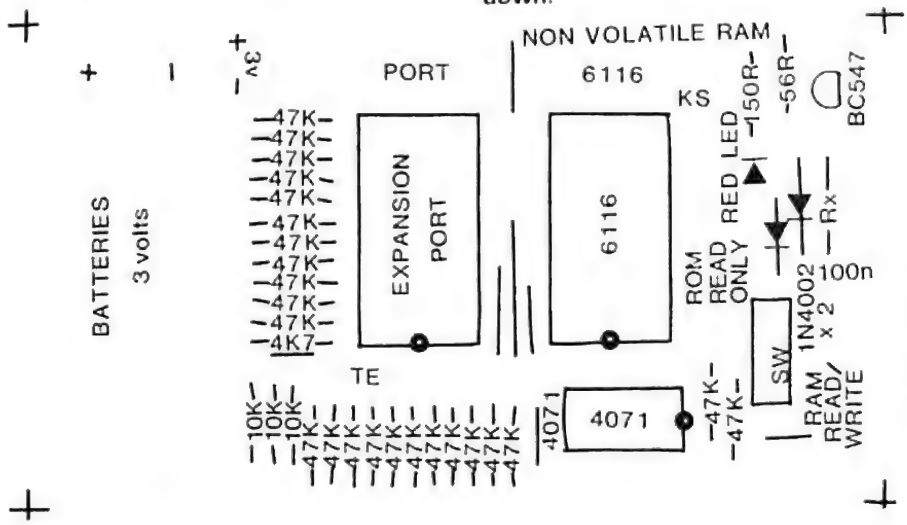
When inserting and removing the RAM from the TEC, the RESET button must be pressed. This will freeze the address and data bus and prevent any glitches from entering either the RAM or TEC. Remove and insert the RAM card quickly to prevent excess voltage appearing on the pins of the 6116.

If this does occur, the circuitry inside the 6116 may heat up excessively and cause the TEC to crash. The RAM will also lose its contents, but may not be permanently damaged.

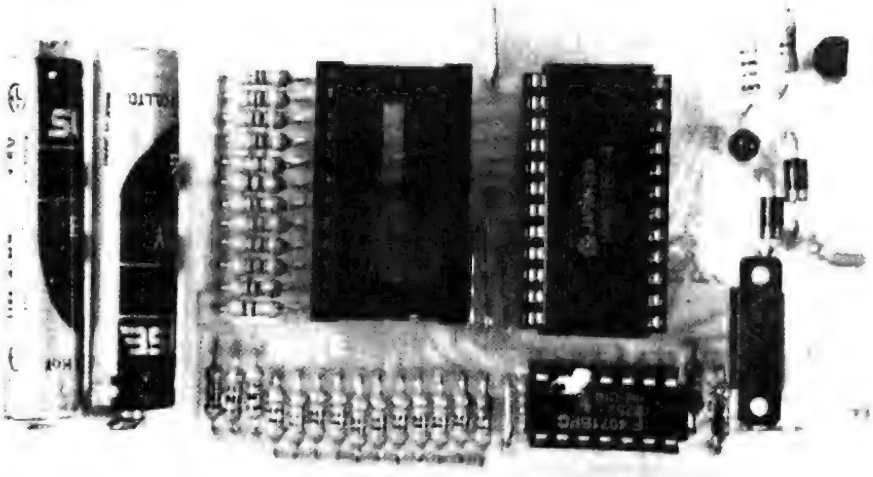
CONSTRUCTION

All the components are mounted on the top of the board in positions as shown by the overlay. The ROM/RAM SELECT is a slide switch and it is best to keep to a slide switch so that the writing on the board reads correctly.

The RAM card is connected to the TEC via a 24 pin wire wrap socket and component header plug soldered together to form a stand-off. The 6116 faces towards the switch as does the 4071 and this may mean the writing on the chip(s) is up-side-down.



See P. 75 for PC artwork.



The RAM card is accessed at the address of the socket in which it is placed. This means it is addressed at **1000** to **17FF** in the expansion port socket. It can also be placed in the RAM socket and is addressed at **0800** to **0FFF**. If placed in the EPROM socket it must already contain a start-up routine for the TEC and is addressed at **0000** to **07FF.**

When placed in the expansion port socket, it can be addressed at higher values by cutting pin 18 of the wire-wrap socket and taking a lead to one of the Chip Select pins near the edge of the board. The lowest of these is addressed as **1800** the next as **2000** and the next as **2800** etc.

If the RAM card is used in the monitor socket it can only be used in the READ MODE as this socket does not have a READ/WRITE line.

This situation also applies when using the RAM card in our dedicated computer project as described in the book: **ELECTRONICS FOR MODEL RAILWAYS.**

The RAM CARD can also be used to create programs for the Microcomp. Any of the programs described in the Microcomp article can be typed into the RAM and executed on the 'comp.

Remember, the Microcomp programs are designed exclusively for the 'comp as it has only a single output latch - the TEC has two output latches.

DUMP ROUTINE

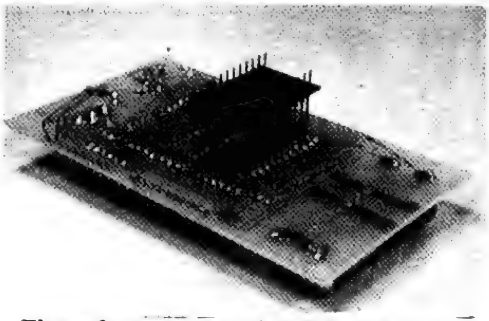
You can use the RAM CARD for many other applications and also transfer up to 2k of program into the card by loading the following into the TEC at **0800:**

```
        TO  11  00  10
      FROM  21  XX  XX
No of BYTES 01  YY  YY
            ED  B0
            C7
```

Where **XX XX** is the start of the program you wish to copy and **YY YY** is the number of bytes you wish to transfer. e.g: If the program to be copied is at **0900** and 80 bytes long, the program at **0800** is:

```
11  00  10
21  00  09
01  80  00
ED  B0
C7
```

Note: If the program starts at **0900** and finishes at**090F**, you must insert **01 10 00** into the program because 0900 to 090F contains 16 bytes of program (10 hex bytes) Not **0F** bytes!



The wire-wrap and DIP header are soldered together to form a stand-off.

If you have written the program at **0800**, you can place the DUMP ROUTINE at **0900**. This is how to do it. Load the program, go back to the start of the program (0900) and push GO. Do not push RESET. The dump program will then be executed. The dump routine can be placed anywhere in RAM by using this method.

Suppose you want to copy the MONITOR ROM. Load the following Dump Routine into the TEC at 0800:

```
11  00  10
21  00  00
01  FF  07
ED  B0
C7          Push RESET, GO.
```

Within a fraction of a second the monitor program will be loaded into the CARD. Change the Read/Write switch to READ and the contents will be protected. Remove the Monitor ROM from the TEC. Insert the CARD and turn the TEC on. It will start up as normal.

To remove a program from the CARD, it is best to fill it with FF's. This will be needed in later programming, when you want to transfer a program from the CARD to an EPROM.

The unused locations will contain FF and these can be burnt to any other value. A value such as 00 cannot be 'burnt down'. For more detail on this, see the EPROM PROGRAMMER project.

TO FILL THE CARD WITH FF's:

at 0800:

```
LD BC 07FF      01 FF 07
LD HL 1000      21 00 10
LD A,FF         3E FF
LD (HL)A        77
INC HL          23
DEC BC          0B
LD A,B          78
OR C            B1
JRNZ            20 F7
RESTART 0000    C7
```

TO FILL THE TEC WITH FF's:

The TEC RAM can be filled with **FF's** by loading the following into **0800:**

```
11 FF FF
D5
C3 03 08
Reset, GO.
```

This puts **FF FF** onto the stack and the stack increments downwards to **0800**! - until the computer crashes.

Both TEC and CARD can be filled at the same time by changing the first two lines to:

```
01 00 10
21 10 08
```

Using these programs, any number of locations can be filled, anywhere in RAM. They can be filled with any value such as **BB, CC, or 33** etc.

Programs can be transferred from the TEC to CARD and from CARD to TEC via the DUMP ROUTINE. Some examples are given in the EPROM BURNER article.

IF THE RAM CARD FAILS TO WORK:

There are three major faults which may occur with this project:

1. **You cannot write into memory.**
2. **Information in the RAM card is lost.**
3. **The TEC starts to play up.**

If it is not possible to write directly into the RAM card or dump into it via a DUMP ROUTINE, the fault will lie in the READ/WRITE line. This is pin 21 of the 6116. It must be low to be able to change the data. Test it with a logic probe or high impedance multimeter. Don't forget to address the RAM correctly (via the addresses given previously).

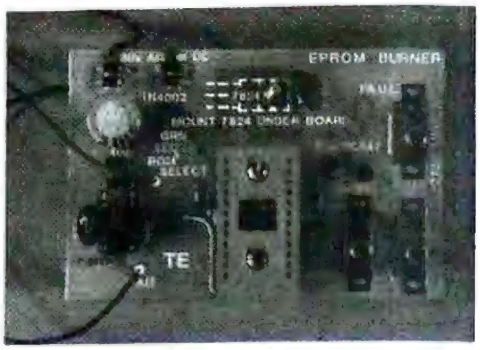
If the information in the RAM card gets lost, the fault will lie with pin 21 of the 6116. It may be floating or go LOW for brief periods so that noise and glitches enter the address and data lines to change the stored data. This problem can also be due to weak batteries or poor contact between the cell and the disk on the bottom of the cell. Try a different brand.

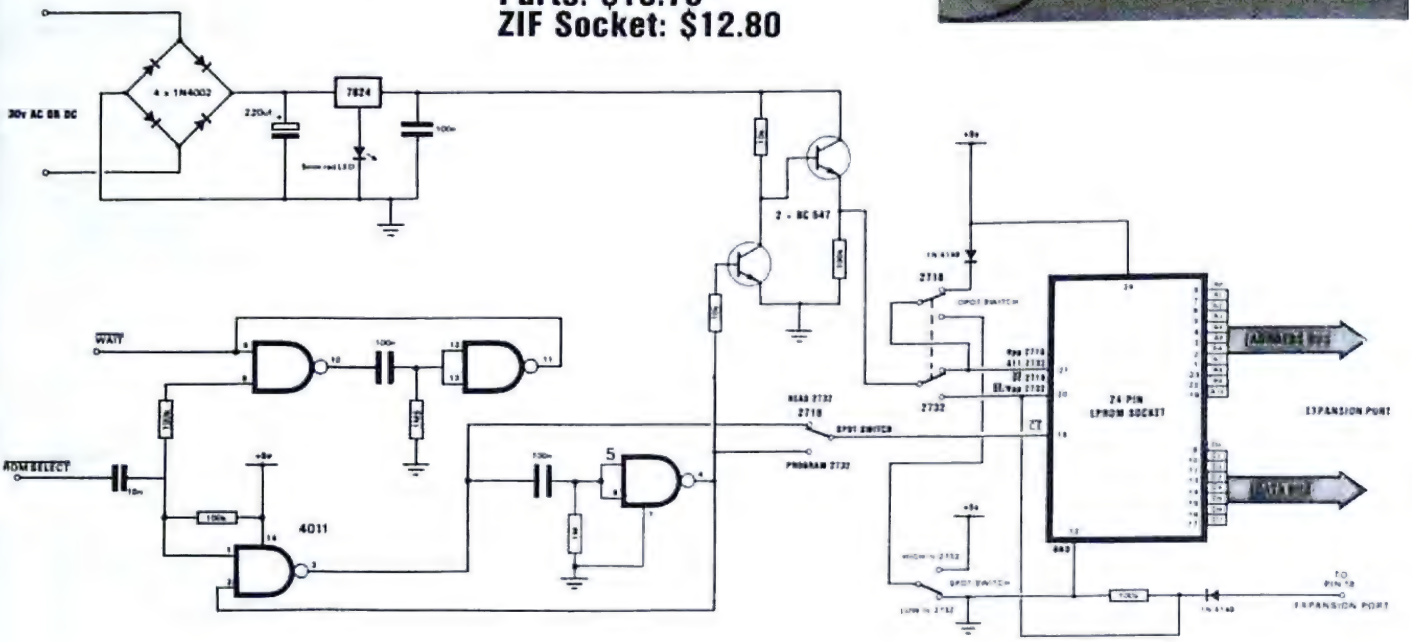
If the TEC starts to play up, it may be due to the RAM card drawing too much current. Feel the RAM chip. If it is getting hot, remove it immediately and let it cool down. The fault may be due to the way you inserted the CARD into the TEC.

Make sure you push the RESET button while inserting the card so that the buses are in a non-active state. If the TEC continues to play up when the card is re-fitted, replace the 4071 and/or the 6116 chip.

EPROM BURNER

PC Board: $3.50
Parts: $13.70
ZIF Socket: $12.80





An EPROM BURNER is the greatest thing to hit the TEC since the regulator was put under the board!

It adds the versatility you have wanted for ages.

To be able to save a program in a permanent form is the final goal of programming.

The TEC RAM CARD and EPROM BURNER combine to make a system capable of generating, testing and producing programs in hard form which can be saved, stored or sold - programs capable of emulating almost any task imaginable.

You can take any project from any magazine or book and convert it to a micro design with a consequent saving in parts, space and cost. Its capability can be increased and its reliability improved by using a tried-and-proven micro design.

By using the NON-VOLATILE RAM as the intermediate stage and the MON 2 monitor (with insert and delete functions) for the production of the program, you can generate, and have running, any machine code program, before burning it permanently into an EPROM.

Burning an EPROM is the final stage and you should be thoroughly satisfied with the performance of a program BEFORE-HAND as it cannot be changed once it is burnt.

This is not entirely true as you can change some values and 'burn-down' any value to zero.

This is an important fact to remember when programming and we will explain what we mean:

EPROMs are purchased in blank form and this means the cells (of which there are 16,384 in a 2716 and 32,768 in a 2732) do not hold any charges of electricity.

Due to buffering circuits in the EPROM, the output from a blank device will be a set of HIGHs. Advantage is made of this as you will see. Eight cells are accessed at a time and if the value is read, it will be 1, 1, 1, 1, 1, 1, 1, 1. But we don't want to read a blank ROM - we want to program it with useful commands and data.

In Hexadecimal notation, the blank EPROM produces FF's from each set of 8 locations. This is called a byte and as we burn each byte in the EPROM burner, we convert the FF's into a lower value. If we don't burn a particular location, its value remains FF.

If we burn all 8 cells, the resulting value will be 00 and the designers of micro-processors (such as the Z-80) have given a very clever command to this value. It is "NO-OPERATION" in which the processor glides over the location with-out affecting any of the remaining program.

PARTS LIST

2 - 10k
3 - 100k
1 - 1M
1 - 1M5

1 - 10n greencap
3 - 100n
1 - 220uf 35v electro

2 - 1N 4148 signal diodes
4 - 1N 4002 power diodes
1 - red LED

2 - BC 547 transistors
1 - 4011 IC
1 - 7824 regulator

1 - 14 pin IC socket
1 - 24 pin wire-wrap socket
1 - 24 pin DIP header
1 - 24 pin ZIF socket 12.80 EXTRA

2 - 10cm hook-up flex
2 - matrix pins (for TEC)
2 - matrix pin connectors
4cm heat-shrink tubing
1 - 6BA nut and bolt
3 - DPDT slide switches

1 - EPROM BURNER PC BOARD

The advantage of having a No-OPeration command as **00** means any location can be 'burnt-down' to 00 if it is required to be removed.

This is where the term 'burn-down' comes from. Whenever an EPROM is burnt or programmed, the value produced is less than the starting value for the location.

We said values cannot be changed once burnt, but in some cases you can reduce the value if the following conditions are met:

The main criteria is: **the cells you wish to change must be 1's.**

The table below shows the values which can be burnt down and those which cannot.

BURN-DOWN TABLE:

	F	E	D	C	B	A	9	8	7	6	5	4	3	2	1	0
ANY VALUE		C A 8 6 4 2 0	C 9 8 5 4 1 0	8 4 0	A 9 8 3 2 1 0	8 2 0	8 1 0	0	6 5 4 3 2 1 0	4 2 0	4 1 0	0	2 1 0	0	0	0

The bold value can be burnt down to the values shown in the column.

Values cannot be 'burnt-up' as we cannot produce HIGHs in an EPROM BURNER.

The only way we can restore HIGHs or 1's to the cells of an EPROM is to put it under an ultra violet light source, whereby ALL the locations will be erased and converted to 1's.

THE CONCEPT

Locations in an EPROM can be burnt if a voltage of 25v is applied to the Vpp pin and pin CE pulsed for 50 milliseconds.

The cells which will be given a charge of electricity will depend on the address which is being accessed and the value of data present on the data lines.

These are the only requirements and programming can be done with a simple set of switches. Unfortunately this would take an enormous length of time as 11 switches would be required for the address lines 8 switches for the data lines and each would have to be set for each byte of information.

A 2716 contains 2048 bytes and if a byte is burnt incorrectly, the whole procedure would have to be repeated.

The other inconvenience is all the bytes in the program would have to be converted to binary so that they can be loaded via the switches.

All this would take so long that the operation of burning would become a head-ache.

By using the TEC, the address values increment automatically as the program advances and the values of data are automatically converted to binary when each location is being burnt. This means you can program in Hex.

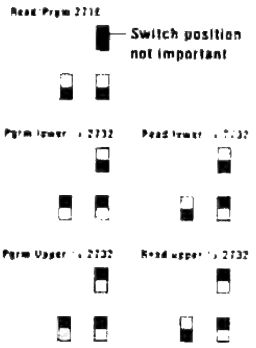
In this way an hours' work is converted to only a few minutes.

This is the function of an EPROM BURNER. It connects an EPROM to the address and data buses, provides the necessary 50 millisecond programming pulse and the 25v supply.

To hold the data steady on the data bus for the duration of the burn, it is necessary to HALT the computer. This is achieved by using another monostable connected to the WAIT line, with a pulse length which is slightly longer than 50 milliseconds.

When you think of it, 50 milliseconds is 20Hz and when you include a short additional delay for the wait function and a number of machine cycles for the execution of a program to carry out the burn operation, you arrive at a burn rate of about 15 locations per second.

Divide this value into the number of locations you wish to burn and you arrive at the length of time for burning an EPROM. That's why it may take a minute or so.



Switch positions for programming and reading 2716's and 2732's

HOW THE CIRCUIT WORKS

The circuit is very simple and consists of a number of building blocks which come together via the ROM SELECT line from the computer.

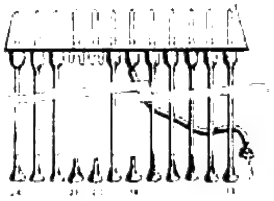
Starting at the top of the diagram, the 25v is derived from a 7824 voltage regulator which has been 'jacked up' by 1.7v by the inclusion of a red LED in the COMMON line. This gives and output of 25.7v and by the time it reaches the EPROM, a voltage drop of .5v has occured across the switching transistor. This transistor is switched via the output line of the 50 millisecond monostable.

The 25v line need not be switched ON and OFF when programming but must not be present when the EPROM is to be removed from the socket. By switching the voltage as we have done, the EPROM can be removed without damage.

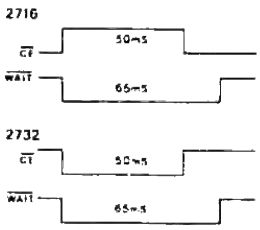
In this article we have included only a very simple burning routine which you can load into **0800** and get the project working.

The final two circuit blocks are mono-stables or one-shots, created from NAND gates. The lower monostable produces a 50 millisecond delay and the upper a 65 millisecond delay.

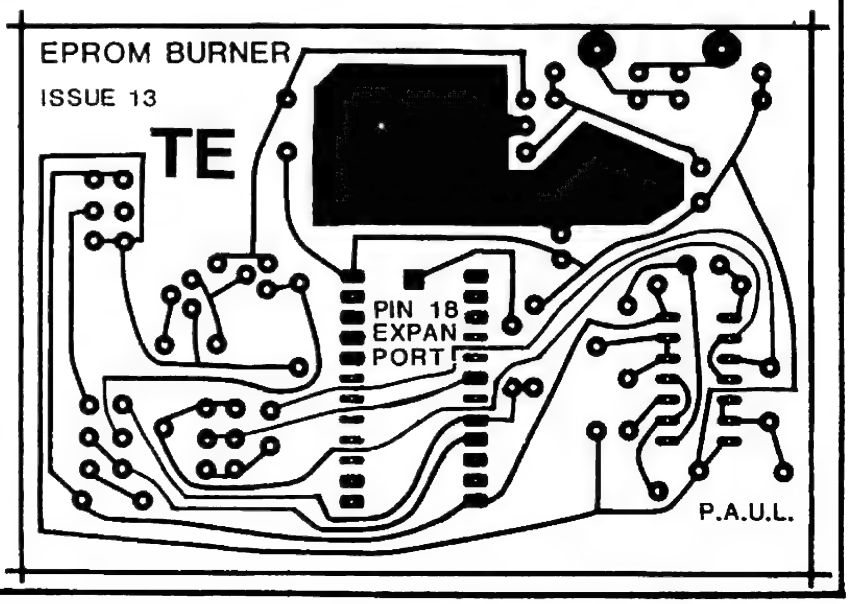
It places data on the data lines, turns on the required address lines and turns on a Chip Select line (located near the edge of the TEC PC).



Don't forget to add the jumper lead from pin 18 of the wire-wrap to the PC board and cut pins 18, 20 and 21.



The programming pulses differ between the 2716 and 2732.



The timing diagrams on the previous page start at the commencement of the program (see page 48 for the program). The program runs for the first 4 lines and in the fifth line the instruction is to load the contents of the accumulator into the location pointed to by the DE register pair. This instruction does three things; not necessarily in this order:

It places data on the data lines, turns on the required address lines and turns on a Chip Select line (located near the edge of the PC).

This line accesses either a RAM or ROM chip connected to it and the address starts at 1800.

This action triggers both monostables and the WAIT monostable immediately goes LOW to HALT the computer.

The program is stopped and the EPROM BURNER circuit takes advantage of the data appearing on the address and data lines. This address is loaded into the EPROM and the data placed on the cells at this particular location.

The high voltage is turned on for 50 milliseconds and at the same time CE is pulsed. This permanently puts the value of data in the EPROM.

The 50 millisecond monostable ends it timing cycle and 15 milliseconds later the WAIT monostable goes HIGH. This enables the computer to continue through the program and come to a JUMP RELATIVE instruction to bring it back to line 5.

This time DE will be pointing to the next higher location and A will contain a new value of data. The count-register-pair BC will be one less than previously. The program continues to loop until BC pair is ZERO.

At the conclusion of the burn routine the program jumps to address 0000 and the monitor program is executed to bring 0800 on the screen with the reset beep to indicate the end of burn.

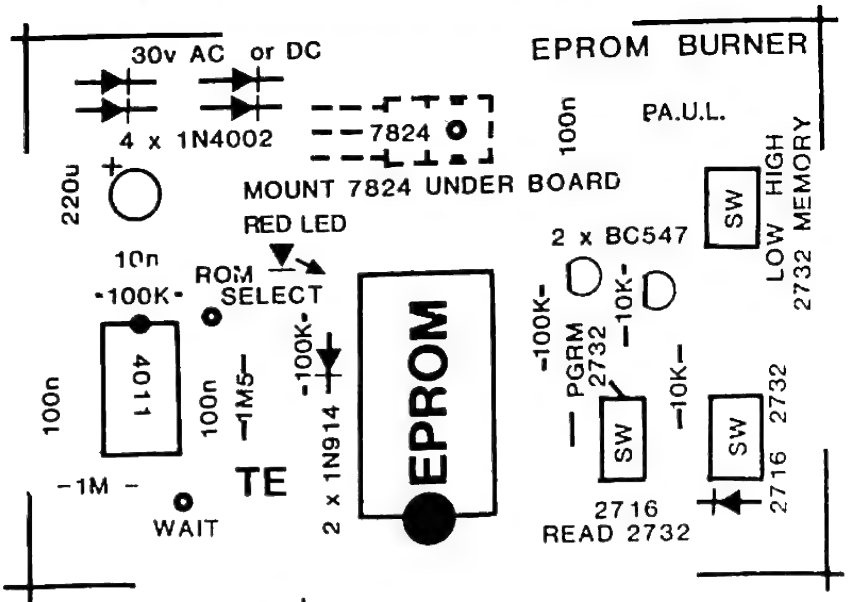
RECAP:

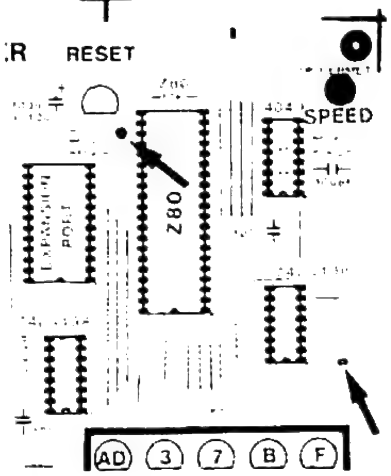
The program burns the EPROM at address **1800 - 1FFF**. To look at the data in EPROM: address **1000 - 17FF**. When the EPROM is removed from the programmer, its address will depend on the project you are using it in, but more than likely it will be the only programmed chip and thus it will be **0000 - 07FF** for a 2716 and **0000 - 0FFF** for a 2732.

CONSTRUCTION

Begin with the resistors and signals diodes. Keep them close to the board before and after soldering but be careful not to damage them with heat. Four power diodes are needed for the 30v bridge. Fit them next, along with the 10n and 100n greencaps.

There is one jumper link on the PC board and this can be made from a lead cut from one of the components.





The Chip Enable pin and Wait pin on the TEC.

The 7824 voltage regulator is mounted under the board and fixed to it with a nut and bolt. The pins fit through the holes provided and are trimmed on the topside of the board.

The two transistors and 14 pin IC socket are the next to the be added and then the red LED.

The 220uf filter electrolytic must be mounted around the correct way and the board is ready for the hardware parts.

Push the 24 pin wire-wrap socket through the holes in the PC and carefully solder it in position. **Cut off pins 18, 20 and 21.**

A short jumper goes from pin 18 of the DIP header to a solder-land on the PC board (located between pins 12 and 13). The easiest way to add this jumper is to solder one end to pin 18 of the DIP plug and the other end to the PC board BEFORE soldering the DIP plug to the wire-wrap socket.

Keep the pins of the wire-wrap full length and solder the 24 pin DIP header into position.

The three slide switches are mounted through the board if the holes are large enough. But if only small holes have been drilled, the legs will need to be extended with short lengths of tinned copper wire and the switches mounted above the board.

Finally connect two jumper leads for the WAIT and ROM SELECT lines and a length of twin flex for the 30v line.

The jumpers require matrix connectors and a short piece of heat-shrink tubing over them to make them sturdy. Heat the tubing with a flame to make it shrink over the connector.

The twin flex requires a 2 pin DIN plug so that it will fit the 30v socket on the TEC POWER SUPPLY.

Fit the 4011 IC and the board is ready.

Two matrix pins will be required on the TEC PC board to take the jumpers. These are soldered as shown in the diagram opposite and are included in the kit.

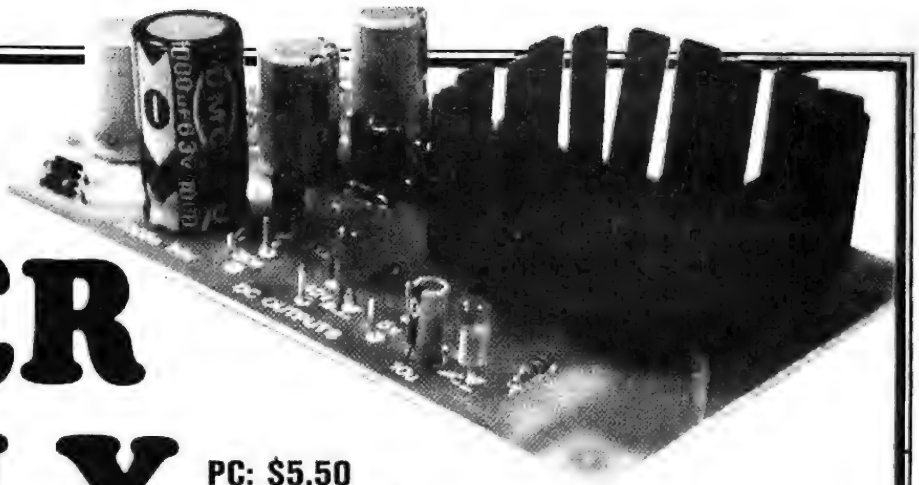
If you intend to produce a number of EPROMs you will need a ZIF socket. These are expensive (too expensive), but are essential if you want to avoid the damage caused by constantly inserting and removing EPROMs from an IC socket. The wire-wrap socket will accept about 50 insertions and removals before it gets a little weak. If the pins do not make good contact, the wrong values will be burnt.

When fitting a ZIF socket, push it firmly into the wire-wrap by starting at one end and gradually introducing the pins, two at a time. You cannot force all the pins together at once.

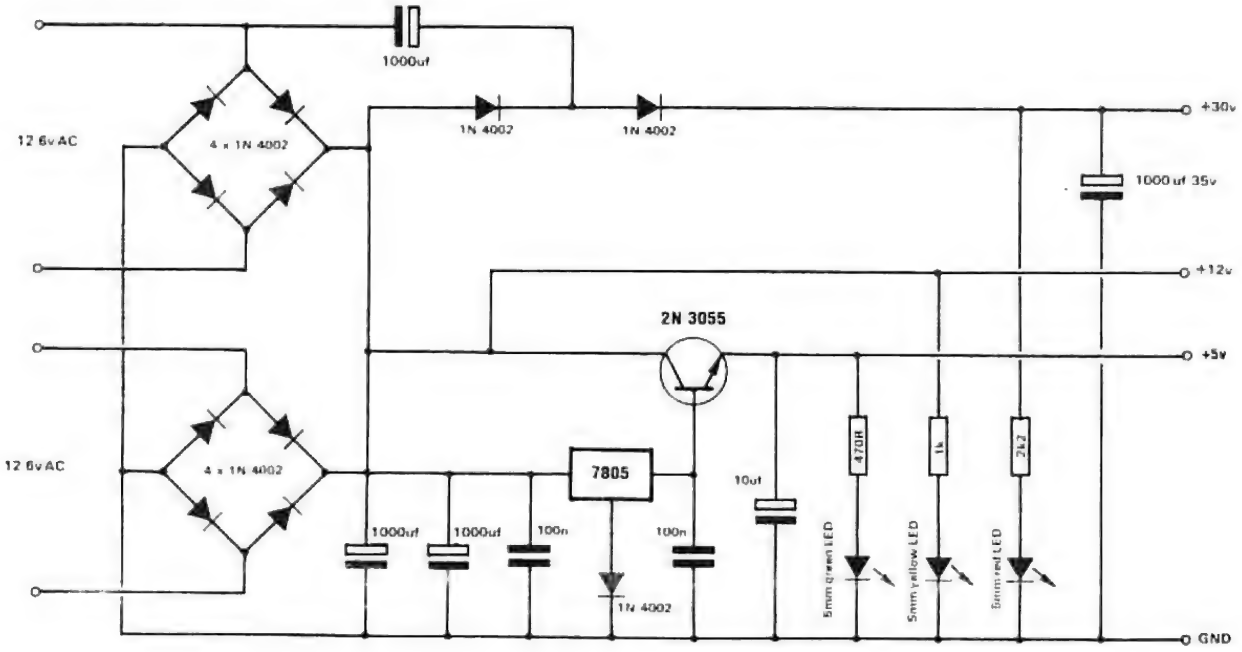
Pin 1 of the EPROM is towards you and this means the lever of the ZIF socket is also towards you.

Cont. P. 48. . .

TEC POWER SUPPLY



PC: $5.50
Parts: $13.90
Complete Kit incl. Transformer, PC & case: $44.25



With the gradual expansion of the TEC, we have come to the stage where we have run out of voltage and current from a plug pack.

A 12 volt 500mA plug pack may sound ideal in theory but when you connect it to a project requiring about 300mA, a cruel thing happens. The output voltage falls from 12v to 10v!

This may be ok if you are dropping it down to 5v via a regulator, but when you want the full 12v for say a voltage doubling operation, 10 volts is not enough!

For too long we have been gulled into believing the ratings of transformers and plug packs. It's only when we require the full rated output that we realize it will not produce.

We learnt our first lesson with the 2155 in a power supply some years ago. Its stated output is 15v AC at 1 amp and this really means 1 amp AC. It also has an AC rating of 15VA and this is very similar to saying 15 watts.

But as soon as we place a 2155 in a power supply we convert the AC to DC via a bridge and gladly accept the output rise to about 21v DC, which is about 40% higher than the AC voltage.

Since the volt-amp rating of the transformer is a CONSTANT (a **constant** is a value in a formula which does not alter) and is 15VA, we must derate the output current to 700mA to maintain the rating of 15VA (or 15 watts).

Thus we can safely draw only about 700mA from a 2155.

There are further projects being designed for the TEC and they include a VDU, possibly for the next issue. The VDU board takes about 350mA, making a total very near the maximum for a 2155 and above the capability of a 500mA plug pack.

We also have a Relay Driver board requiring 12v-15v for the relays and an EPROM BURNER, in this issue, requiring 30v.

All this has led us to design a power supply capable of delivering these 3 voltages. At a later date it can be expanded to deliver about 1.4 amps to the 5v line, to cater for fully expanded TEC's.

The TEC Power Supply is not only for the TEC, but will also power any other project requiring one or more of these voltage.

The project is mounted in a neat plastic case as supplied by Altronics and Dick Smith and is the LARGER of the two (in the range).

These cases make the project look very professional and can accommodate both 2155's and the PC board. The floor of the case has a number of spigots for mounting the board and transformer(s) so that everything fits firmly and neatly in position.

As with all projects which involve connection to the mains, this project must not be connected until it is checked by someone with experience.

The first question you will ask is "Why use two 2155's and two bridges."

The answer is simple. Working with currents up to 1 amp produce very few problems. Components such as diodes, regulators and transformers are designed for a maximum of 1 amp. When you go over 1 amp, the problems start.

Nothing is designed for 1.5 amp or 2 amp and in fact high power components start at 3 amp.

This means a 1.5 amp or 2 amp power supply falls into the middle of component availability.

The cheapest and best solution is to produce two 1 amp supplies and parallel them up. This is what we have done. Two 2155 transformers are taken to two bridges and from there the current gets divided between the three outputs. Most of the current will be required by the 5v line while the 15v and 30v lines will not require heavy currents.

In practice, only about 30-50mA will be required on the 30v line for the EPROM BURNER and only about 100mA for the Relay Driver board.

HOW THE CIRCUIT WORKS

The circuit is basically a 7805 regulated power supply with a series pass 2N 3055 transistor to supply the current for the 5v rail.

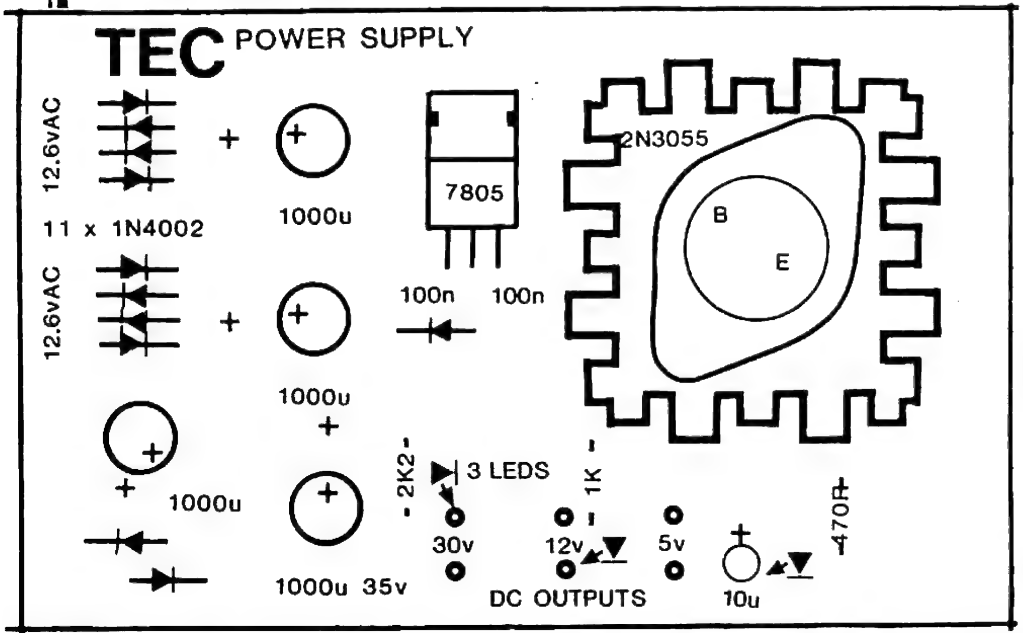
The 15v rail is taken from the input to the regulator and is filtered and smoothed DC, but not regulated.

Referring to the circuit diagram, the 30v rail is obtained by voltage doubling the 15v line. The top bridge is responsible for this and means it is the bridge which operates when only ONE transformer is used. The lower bridge provides the back-up when more than about 700mA total is required.

The 5v line has some interesting points.

The output transistor is an emitter follower in which the base voltage is determined by the output voltage of the 7805 regulator. As we know, the emitter of a transistor in an emitter follower arrangement is .6v lower than the base. Thus, to obtain a 5v output, we must supply the base with 5.6v. This is achieved by placing a diode in the 'common' line of the 7805 and increases the output by .6v. Thus it emerges from the 2N 3055 at exactly 5v.

In this arrangement we have lost the shut-down facility of the 7805. But since we have found this to be very unreliable with a 2155 transformer, nothing has really been lost.



The overlay for the TEC POWER SUPPLY. The 7805 voltage regulator does not require a heatsink but the 2N 3055 must be suitably heatsinked as it is the current regulating component.

PARTS LIST

1 - 470R ¼watt
1 - 1k
1 - 2k2

2 - 100n greencap
1 - 10mfd electro
3 - 1000mfd 16v or 25v
1 - 1000mfd 35v or 63v

11 - 1N 4002 diodes

1 - 5mm red LED
1 - 5mm green LED
1 - 5mm yellow LED

1 - 7805 regulator
1 - 2N 3055 transistor

1 - TO-3 heatsink

2 - 4BA nuts and bolts
3 - 6BA nuts and bolts

10 x 20cm hook-up flex
1 - 2 pin DIN socket
1 - 3.5mm socket
1 - RCA socket

1 - 2155 transformer
4cm heat-shrink tubing
1 - cord clamp
1 - power cord and plug-top
1 - solder tag for earth lead
1 - H 0482 case
1 - SPDT switch for mains
heatsink compound

1 - TEC POWER SUPPLY PC

Extra parts: 1 - 2 pin DIN plug, 1 - 3.5mm plug & 1 - RCA plug: $1.30

The 7805 requires about 1.5 amps to be flowing before it will shut down and this current is not available from a 2155.

When looking at the photo of the completed project you will notice the regulator does not have a heat fin. This is because the regulator does not supply the current to the 5v line. The transistor does all the work. The regulator supplies a voltage and about 50-80mA to the 2N 3055, to drive it. The transistor has a current gain of about 10 to 20 and thus it is capable of supplying about 700-1400mA. This is why the 2N 3055 must be properly heat sinked.

CONSTRUCTION

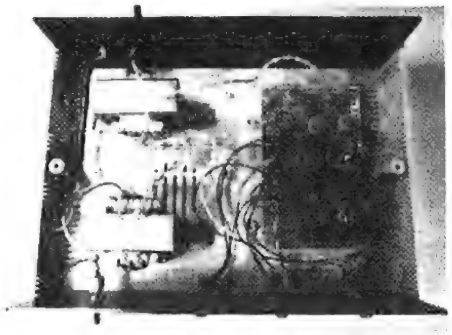
All the comonents are mounted on a single PC board with flying leads to each of the output jacks and the indicator LEDs. If using one transformer, a twisted pair goes from the 12.6v AC holes on the middle of the edge of the PC to the 0v and 12.6v tapping on the transformer.

Start by fitting the 11 power diodes. The cathode end is identified via a white band around one end and this corresponds to the line on the symbol on the overlay.

The diodes must be pushed home BEFORE soldering and must touch the board AFTER soldering. This is necessary as the copper tracks are designed to act as a heatsink to prevent the diodes getting too hot.

Next fit the three resistors. These are current-limiting resistors for the indicator LEDs.

Fit the two 100n greencaps and the 4 electrolytics. Note the marking on the electro's indicates the negative lead whereas the board identifies the positive lead. Don't get mixed up.



Layout of 2- 2155 transformers and PC board inside case.

compound squeezes from around the edges of the transistor. If you have done this correctly, it will be even all round. Solder the base and emitter leads. The collector is the case of the transistor and gets its voltage via the bolts. That's why they must be screwed up tightly and don't get the thermal compound on the bolts.

The next stage is to attach the flying leads to the board for the input and output.

It is suggested that a colour code is used so that each output can be recognised by a colour. This will prevent a major mistake.

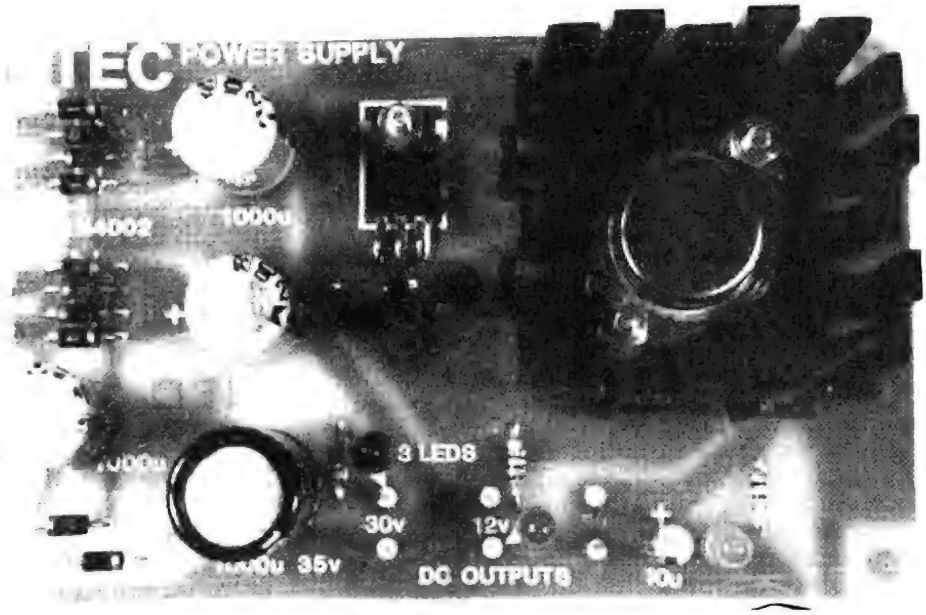
The 3 output sockets are different to each other so that the plugs must also be different. Each project you connect to the supply must be fitted with the correct type of plug and this will prevent the wrong voltage being selected.

The sockets we have chosen are RCA for the 5v line, 3.5mm for the 15v and 2 pin DIN for the 30v. The RCA socket is used for the 5v because it provides the greatest amount of contact between plug and socket for the higher current flow.

These sockets are mounted on the front panel along with the indicator LEDs.

The exact position for these sockets is not critical except you have to make allowance for the heat fin and transformer. This restricts the layout somewhat and the photo shows a suitable positioning.

The only other component on the front panel is a 240v power switch.



Next push the leads of the 7805 through the hoes in the board and bend the regulator over. Attach it to the board with a nut and bolt and then solder the leads.

The final component to mount is the power transistor.

Place the heat-fin on the board and before the transistor is fitted into place, smear a little thermal compound on the underside of the transistor.

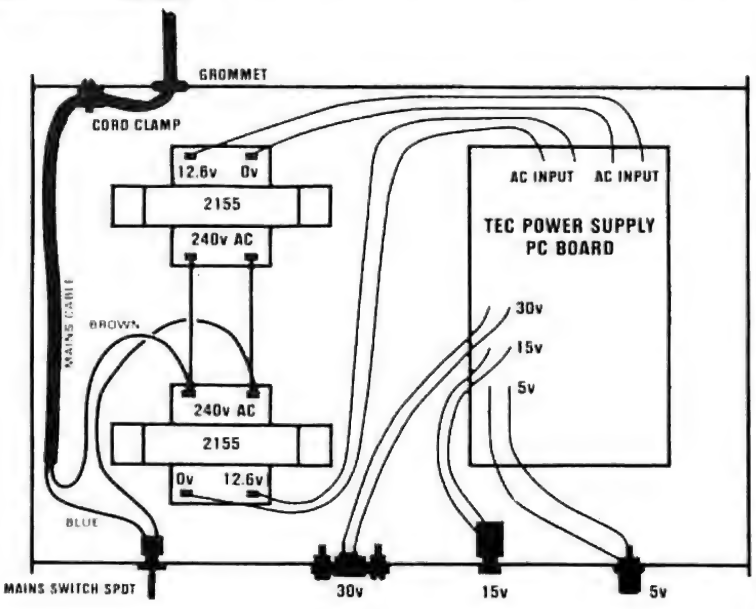
Take care not to short the base or emitter leads against the heatsink.

Place the transistor onto the heatsink and attach to the board with two nuts and bolts. As you tighten the bolts, you will notice the

2 pin DIN for 30v

3.5mm for 15v

RCA for 5v.



Layout for 2 transformers. If only one transformer is required, the smaller case can be used.

WIRING THE MAINS SECTION

Wiring the mains lead to the power supply is simple enough except that it must be done according to RULE. This states that the lead must be attached or 'gripped' to the cabinet so that it will not pull out or pull on the wires inside the box.

Make a hole in the rear panel which is just large enough and push 20cm of the lead through. Place a cord anchor about 2cm from this hole and fasten the lead to the cord grip with a nut and bolt.

Remove 9cm of the outside sheath and separate the three inner wires. The ACTIVE lead is RED or BROWN and must be taken to the switch on the front panel. Bare ½cm of the lead and slip a 2cm length of heat-shrink tubing over this lead. Solder to either terminal of the switch.

Cut the **BLUE** or **BLACK** neutral wire slightly shorter and repeat the same procedure, this time connecting to either of the primary terminals on the transformer. Don't forget the short length of heat-shrink.

Finally the earth lead must be connected. Surprisingly, this lead must be the longest so that it is the last lead to be broken, should the cord be pulled. Solder this to a solder tag and screw it onto one leg of the transformer.

A short jumper is required between the switch and the other input of the transformer. Add this and include the heat-shrink.

The heat-shrink must now be pushed over the exposed parts of each of the connections and shrunk into place by heating with a match or candle. The object of the exercise is to cover all the mains wiring so that nothing can touch a live wire.

If you have bought a MOULDED PLUG, the power cord will now be complete. If the plug-top requires attaching, use the colour code given above for the correct connection of the wires. You may think the choice of colours is rather poor. Don't blame me. It's an international code, to assist colour-blind people.

TESTING THE POWER SUPPLY

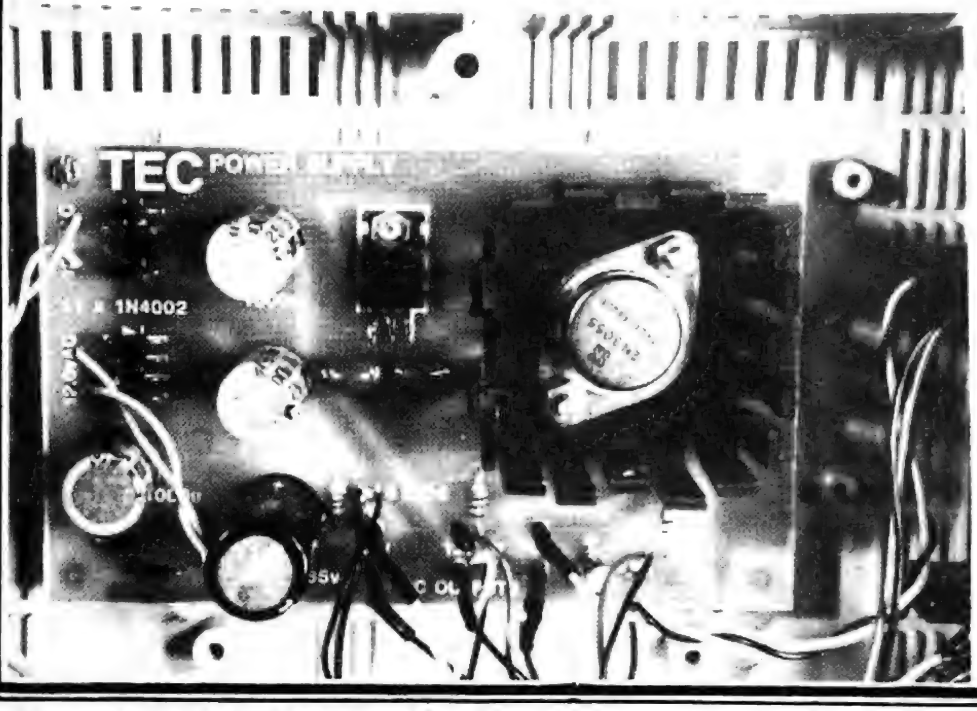
Testing the power supply is done in two stages.

Connect the plug to the mains and turn the switch on the case ON and OFF very quickly. The three indicator LEDs should come on. If they do, you can be sure the three voltages are present.

Next you must test the supply to see if any of the components are going to overheat. This is always an important stage in testing a project which has the potential for supplying a lot of power.

Turn the supply ON for 10 seconds, then pull the plug from the wall. Feel the temperature of each component. If everything is cool, repeat for 30 seconds. Again remove the plug and feel each item.

If everything is ok, switch on for 5 minutes and try again.



Parts can be damaged very quickly when in a high current situation like this and if an electrolytic is around the wrong way, it will very quickly heat up and may even explode.

The next stage is to measure the output voltages with a multimeter. The most important voltage is the 5v rail. It must be exactly 5v. TTL projects such as the TEC computer will be connected to this rail and it must be spot on 5v.

The other rails are not quite so critical as they are used to operate relays; and the 30v rail will be taken to a 24v regulator for feeding into a 25v line as shown in the EPROM BURNER project.

Even so, you should make sure they are delivering the required voltage. If the 30v rail, for instance, is only 15v or 20v, something is wrong. And if it is 25v, it will not be sufficient for the EPROM project.

A low voltage, like this, may be due to low mains or a diode missing (not working) in the bridge.

One way to test the diodes is to fully load the supply, via say the 15v rail, and feel the temperature rise of each diode. They should all get equally warm.

If your area suffers from low mains voltage, you can increase the voltage of the supply by connecting between the 0v and 15v tapping on the transformer.

FINAL ASSEMBLY

After testing the power supply for temperature rise, fit the front and back panels and close the case using the two long screws provided. Use white lettra-set or other form of identification to show the three output voltages and the ON position for the mains switch.

POWERING THE TEC

The TEC can be powered from either the 15v line or the 5v line.

If powering from the 15v line, a lead can be taken from the AC terminals on the TEC to the 15v output. The 7805 regulator on the TEC will provide the regulation. It is interesting to note the input lines can be either way around as the TEC has a full wave rectifier and this means the input voltage can be of any polarity.

If supplying from the 5v output, things are different. You must connect to the 5v rail of the TEC. This is done by connecting one lead to the earth line and the other to the output pin of the 7805.

KITT SCANNER

PC Board $1.80
Parts: $5.60

A REALISTIC SCANNER FOR KNIGHT RIDER MODELS.

-by Ken Stone.

Most people are familiar with the adventures of a certain black Trans-Am with 5000 Megabit memory.

This car started life as emotionless and argumentative but by the time it smashed its way through the first episode, everybody wanted to own it.

MPC released a model of the Knight 2000 in 1983 and the first shipment sold out before I could get my hands on one. Thirty shops later . . success!

The MPC model is moulded in black plastic with dark tinted windows and a few chrome parts. The car is supplied with a red tail light but unfortunately the scanner is part of the black moulding.

To make the model more of an attention-getter, I decided to put a working scanner into it.

This project is the results of my design.

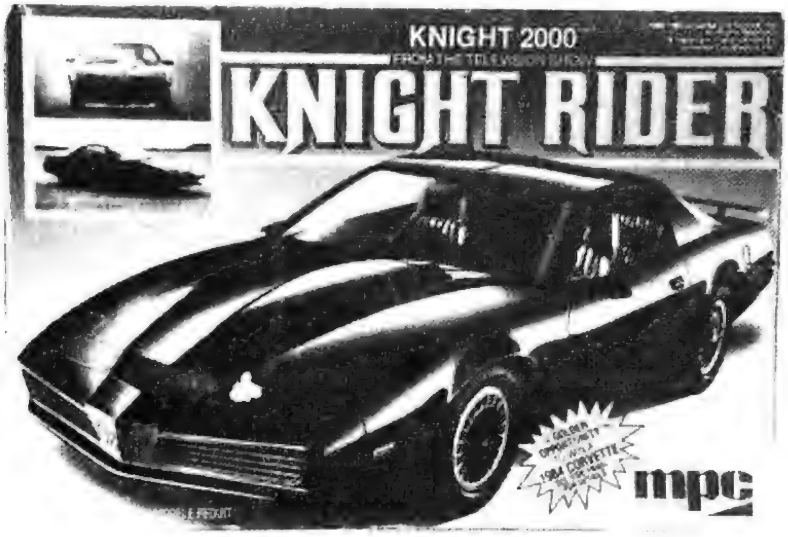
The scanner is made on a small PC board and mounted under the bonnet so that the LEDs shine through a piece of red plastic glued in place of the black plastic moulding.

The turbo V8 engine, drive shaft and exhaust system must also be removed to make room for the PC board.

The first comment you may make is KITT has 8 lights and the scanner only 6. We could only design a simple circuit for 6 LEDs and fortunately only 6 would fit.

HOW THE CIRCUIT WORKS

The circuit consists of two building blocks. The first is a square wave oscillator made up of two transistors in a multivibrator arrangement and the second is a CD 4017 decade counter IC.



The multivibrator contains two extra components to speed up the wave-form and make it acceptable for all brands of 4017's. Unless the output has a very fast rise and fall characteristic, some 4017's fail to operate properly. They either do not work at all or jump two or three outputs, losing the scanning effect.

The two speed-up components are the signal diode and 39k resistor and are essential for reliable operation.

The output of the multivibrator feeds into the CLOCK INPUT of the chip. From there is goes to a complex counting circuit inside the 4017. The circuit counts to 10 and only one output is active (HIGH) during each of the 10 steps.

Initially output pin 3 is HIGH while all others are LOW. After one clock cycle the second output (pin 2) goes HIGH while all others are LOW. After the next clock cycle pin 4 goes HIGH etc through to the 10th output which is pin 11. We could see the effect of these outputs going HIGH by placing 10 light emitting diodes on the lines. They would give a 'running light effect'. Remember this.

We have placed 10 signal diodes on the output of the chip so that we can illuminate a set of Light Emitting Diodes from one of two different lines.

Eight of these diodes form four OR gates to direct the appropriate outputs of the 4017 to the corresponding LEDs. The remaining two diodes equalise the brightness of the two ungated LEDs.

The first 6 outputs operate the diodes in a normal running light sequence. The clever part comes with output Q6. It is taken to the 5th LED and this creates the effect of reversing the

sequence. Q7 drives the 4th LED and this continues the reverse effect until the second LED. The chip has now completed one cycle and the first output is now turned ON. This illuminates the first LED to complete the full back-and-forth scan.

Each time the 4017 goes through its run of 10 outputs, the scanner completes one forward and reverse scan.

This is how the effect is generated. It is the first time anyone has used the chip in this way and its shows you don't have to stick to convention.

CONSTRUCTION

The SCANNER is constructed on a PC board which has one end specially shaped to fit into a plastic model and give the LEDs the radius they need for alignment against a piece of red diffusing screen.

This end is shaped before any of the parts are fitted, by cutting the excess from the board with a pair of side-cutters. After this, the board is finished off with a file.

Refer to the photo before mounting the parts to see how and where they are placed. Some fit against the board and others are mounted upright to take up the least amount of space.

Start by inserting the 11 signal diodes. Some of these lay flat against the board while others are almost upright. They way they stand (or lay) depends on the distance between the holes and you will have to fit them as neatly as possible.

The 10 diodes in a row face the same way and the single diode in the oscillator faces downwards.

Next mount the row of 5 resistors and the single resistor which lays flat against the board. The resistors in the row stand on END and it is important to get the values correct. They are clearly marked on the overlay and you can also refer to the layout diagram in the article.

Next solder the input protection diode to the PC board. This diode stands on END.

Next fit the IC socket so that pin 1 identification (either a cut-out or corner missing from the socket) aligns with the dot on the PC board. This will make it easier to fit the IC correctly.

The two transistors are fitted so that they nearly touch the PC board and only a gap equal to the height of a resistor separates them from the board.

The three electrolytics are mounted so that the positive lead goes down the identified hole on the PC board. You will notice the negative lead is identified on the component while the positive lead is identified on the board. Do not get confused!

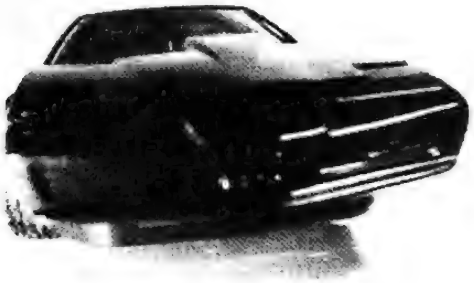
The last components to be mounted are the LEDs.

We have drawn a large diagram to show how these are connected. Once you fit them, they cannot be refitted as the leads will be too short. So get it right.

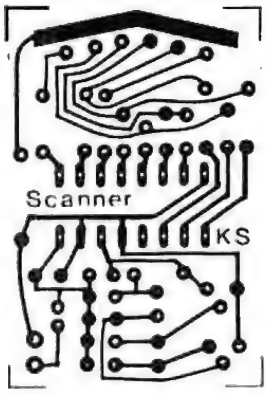
The cathode lead is the shortest and this goes directly to the trackwork on the underside of the board. It does not pass through a hole.

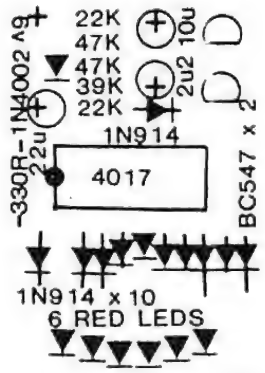
The anode lead passes over the board and down a hole where it is soldered in place. This is an unusual way of mounting LEDs but suits our project perfectly.

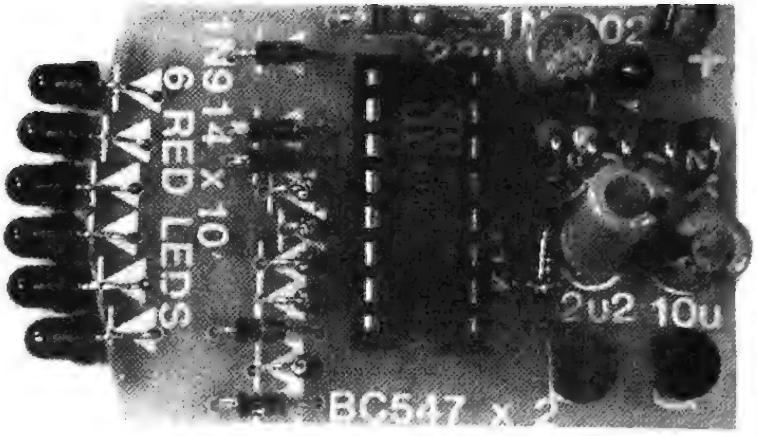
There is one thing you must never do. Never spread the leads of a LED as



SCANNER CIRCUIT







this will crack the plastic encapsulation and it will be damaged. There will, however be some slight spreading of the leads as they are inserted and this is the maximum allowable.

Mounting the LEDS

Cut the anode lead as shown in the diagram and bend it to 90°. Cut the cathode lead very short and the LED is ready for mounting.

Fit it onto the board and solder each lead very quickly to prevent damage to the crystal inside the LED.

At this stage you can test its operation by fitting the IC and connecting the board to a 6v supply. The LED should blink on/off,

If it doesn't, check the orientation of the LED by looking into the body and noting a large 'cup'. This is the cathode and goes directly to the underside of the board. Place a multimeter across the LED, when the project is working. The needle should swing up to 1.5v for a brief period of time.

If it does, but the LED doesn't light, you have possibly damaged the LED, either by separating the leads too far or overheating the LED when soldering.

If the LED does illuminate, continue fitting them until all are placed neatly in a row.

Make sure they continue the radius of the board so that they will fit behind the screen in the car.

Two power leads are soldered to the board and these must be long enough to reach the rear of the model.

The 4 AAA cells are mounted in the boot and soldered together to form a 6v battery pack. A small slide switch is placed under the car to turn the scanner ON.

Now is the time to test the project and watch the effect. You will find it quite hypnotic. If everything works well, mount the board up-side-down in the bonnet compartment so that it is parallel to the ground. Make up a couple of struts to support the board and keep it in position.

Everything is now ready for the final touches of presentation. Complete the assembly of the model and make sure all traces of wiring, batteries and PC board are removed from view.

I know you will be pleased with the effect, but break the news slowly to your friends. Say "Wouldn't it be nice if we had the real effect of the Knight Rider scanner!"
Then flick the switch!

PARTS LIST

1 - 330R ¼ watt
2 - 22k
1 - 39k
2 - 47k

1 - 2.2mfd electrolytic PC mount
1 - 10 mfd PC mount
1 - 22mfd PC mount

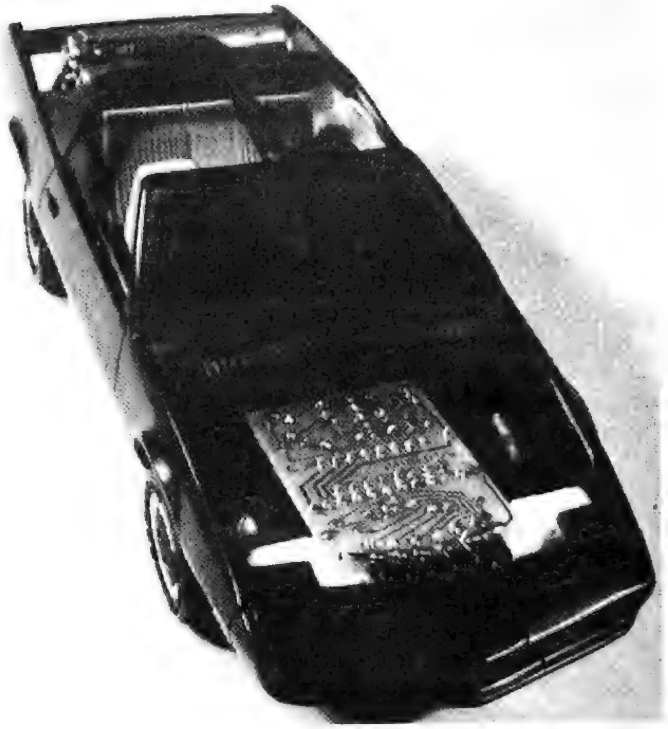
11 - 1N 4148 diodes
1 - 1N 4002 diode

2 - BC 547 transistors
1 - CD 4017 IC

1 - 16 pin IC socket

4 - 'AAA' cells
6 - 3mm red LEDs
1 - miniature switch
1 - piece of red plastic

1 - SCANNER PC BOARD



IF IT DOESN'T WORK

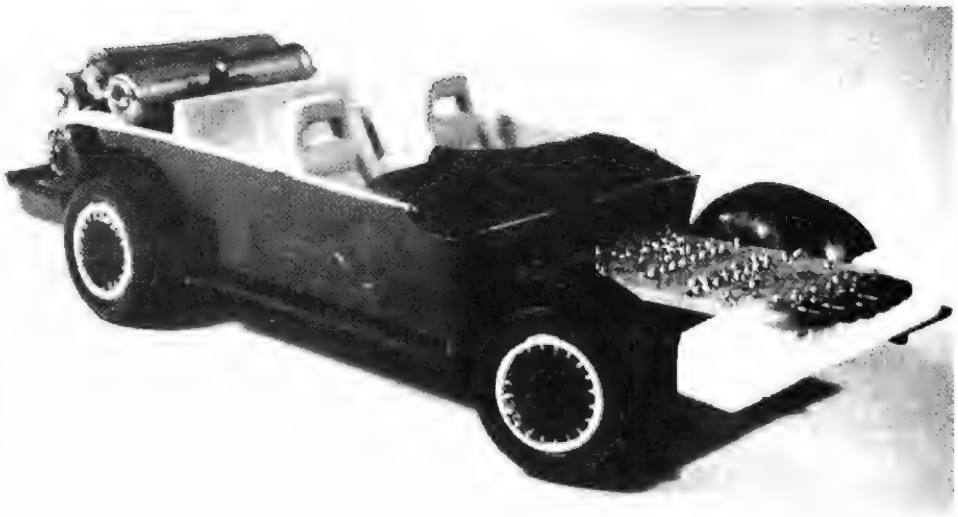
There are two sections to this project. If the LEDs do not scan, the fault will lie in either section. You have to isolate which section is at fault.

Firstly test the multivibrator. Place a multimeter, set to VOLTAGE, between pins 14 and earth. The needle on the meter should oscillate, indicating the transistor section is producing a waveform. If this waveform is present, the 4017 could be at fault. Check the voltage on pin 16. It should be rail voltage. Check the voltage on pins 13 and 15. It must be LOW for the chip to count. If the first LED remains ON, the 4017 could be damaged or the input waveform too low to clock the chip. Try a new chip.

The next stage is to isolate the chip from the multivibrator. This is done by isolating pin 14 from the circuit and connecting a jumper lead to it.

Take this jumper to rail and then to earth. This will produce a full transition on the clock line and hopefully cause the chip to count. If this is successful, the incoming clock pulses are TOO SMALL or of poor quality.

A CRO would be handy to check the waveshape but if this is not available, you can manually clock the chip via the transistor circuit. Firstly take one base lead to ground and then the other base lead to ground. While doing this you can measure the voltage on the collector of the output transistor and note that it changes from LOW to HIGH.



If the chip does not clock, the fault will lie in the output transistor. It may not be connected to earth, the diode may be around the wrong way, or creating a leakage which will inhibit the amplitude of the output pulse. Or the resistor may be the wrong value. The only other possibility is the transistor. It may be damaged and thus will not amplify sufficiently.

If the chip jumps LEDs when scanning, the pulse shaper circuit may be at fault or the amplitude of the signal is too low for the chip. It could also be the 4017. Some 4017's have Schmitt trigger circuits in the input line to reduce the effects of noise. Others do not have in-built Schmitt triggers. Try a different brand of chip.

If one or more of the LEDs fail to come on, place the multimeter on the relevant output and watch the needle. If it pulses HIGH/LOW, the signal is emerging from the chip and the LED will be at fault. (or the soldering). Try the signal on the other end of the diode. If it is not present, the diode is at fault or has been inserted around the wrong way.

If the scanner is running too fast, the electrolytics will be the wrong value or 'dry'. It could also be due to the wrong value of resistance in the base lines.

The 330R resistor in the cathode line of the LEDs is a current limiting resistor to prevent the LEDs taking too much current. Make sure it is connected to the circuit or the LEDs will not come on.

If you follow these suggestions your scanner should be working perfectly.

I hope you find it to be a captivating addition to your Knight kit.

LETTERS...

Here are a few letters from our mailbag. We like to get letters as it gives us feedback and lets us know the success of the projects. It also alerts to any glaring errors or misunderstandings in the articles:

Let's get straight into the letters:

Sir,
A spate of breaking and enterings in our area prompts the thought for a burglar alarm kit. Maybe several different models could be designed. A simple type, a more advanced type such as ultra sonic and an elaborate infra-red type.
I know I will be one of the first to buy a kit as I have had a great deal of success with the other TE kits I have built.

Many thanks for a great magazine,
T Gatfield,
Golden Beach, 4551.

Burglar alarms are big news at the moment. Everyone, from a hardware shop to a locksmith is displaying a wide variety of alarms and deterrents.

Surprisingly, many of these are utterly useless. Some of them can be falsely triggered by aircraft, animals, noises, and trees blowing in the wind.

A recent report on one type being hawked from door to door, stated that the alarm was so unstable that it triggered for no apparent reason. The siren became so annoying that the police issued a noise warning after receiving complaints from local residents.

After repeated service calls from a technician, the conclusion was a poorly designed circuit which could not be rectified.

False alarms like this are fairly common and we quite often hear them wailing in the distance.

This means they offer very little protection, as residents consider it to another false alarm!

We have looked into many alarms and the different principles of operation and decided the most effective type is a silent phone dialler.

It is set-off via a pressure mat, the opening of a window, door or drawer and immediately dials a pre-programmed phone number.

The sensors used with this type of alarm are much more reliable than a sonic system and considerably cheaper.

The Phone Dialler Alarm is one of the 'add-ons' for the Microcomp computer and will be described in the next issue.

It has the capability of dialling a phone number and producing a tone to indicate the alarm has been tripped.

After 5 minutes it will re-dial the number and repeat the tone.

This gives you the choice of dialling the police, your place of employment, or a neighbour, to determine the true situation.

Sir,
I have a number of transistors radios which have ceased to function. I don't know if they have been dropped or left out in the sun. They may be beyond repair or only need a wire attaching.

I have read your servicing articles and would like to know if you have produced one on repairing transistor radios.

I think a lot of readers would have radios lying around like this and would be pleased to get them going. Would a CRO help?

T. Bell,
Howrah, 7018.

In the past we have produced a few notes on servicing transistor radios and have prepared a large article for a future issue.

Repairing radios is a difficult exercise at the best of times and the use of a CRO will not help the situation. It is more likely to muck you up, than help. The trace on the screen will bear no resemblance to the signal you are tracing and it will be very difficult to interpret.

A CRO is not the answer to servicing radios and should be kept to the designing side of audio amplifiers.

Some readers think an ultra-sensitive multimeter will be the answer to servicing but again they can introduce more mysteries than they solve.

Take for instance, a digital meter, when you see a readout such as 6.03v, you get side-tracked by the value and miss the point. Values are not as critical as you think.

The only things you really need are: a cheap multimeter and signal injector. Their use will be described in a future article.

Sir,
In the Egg Timer circuit in issue 6, I have a question on the the operation of the mercury switch.

Take the following situation: When the output of the latch (pin 6) is HIGH, why isn't the chip damaged when the mercury switch is closed? It produces a virtual short to earth and I though this would damage the output.

E. Lee,
Pearce, 2607.

You are correct, the mercury switch will place a short on the output of the 74c14 chip, however this does not cause any damage. There are two reasons for this:

1. Firstly the output of most chips contain a current-limiting arrangement to limit the current to a safe level.

2. The latch circuit contains two diodes (in series) which prevents the output being shorted directly to rail. Also, as the mercury switch closes, the input of the latch will detect the lower threshold voltage and cause the circuit to change before the voltage reaches zero.

PAUL'S TWO-WAY

Paul has built two FM BUGS and uses them regularly to talk to one of his friends two streets away.

By prior arrangement, he organises a time for communicating and holds a two-way conversation, much like a phone call.

He mounts his BUG at one end of the room and tunes in the FM radio to receive his friends BUG. The same arrangement is set up at the other end.

Providing the BUG and radio are far enough apart, the arrangement will not create feedback.

This idea works very well and the ¼ wave antenna transmits over 200 metres in a built-up area without any problems.

This idea opens up possibilities for two-way conversations between home and garage or where normal telephone lines would be impractical.

EPROM BURNER

. . .cont. from P. 22.

The lower switches must be set for 2716 or 2732.

When burning 2716's the upper switch position does not matter as it is not in circuit.

When burning 2732's the upper slide switch selects the UPPER 2k or LOWER 2k of the 2732.

The switch closest to the EPROM is placed in the upper position when programming 2732's and in the lower position to read them.
This switch is placed in the lower position when programming and reading 2716's. Refer to the set of diagrams before carrying out any operation.

The high voltage is derived from a 30v supply. This can be the TEC POWER SUPPLY or from a 30v AC transformer. Very little current is required, however the voltage must not go below 30v or the regulator will dop out. This is because we are generating 25.7v and the regulator requires 3-4v across it for regulation.

Connection of the EPROM BURNER board to the TEC is via a 24 pin wire-wrap and DIP header plug. The board fits in the expansion socket and requires a WAIT line and ROM SELECT line.

The ROM SELECT line is pin 12 of the 74LS138 and WAIT is pin 24 of the Z-80.

Connect these lines to the TEC and plug the EPROM BURNER board into the expansion socket. Connect the 30v supply and the red LED will illuminate to indicate all is ready.

You can burn a new EPROM or blank locations in an old ROM. You can even burn old locations providing they fulfill the requirements mentioned previously.

THE PROGRAM

The BURN PROGRAM can be placed anywhere in the Monitor ROM or typed into the RAM. If placed in the RAM, it will need to be typed each time an EPROM is to be burnt.

The program is very simple and does not have any checking facility to prevent it burning over previous program.

The absence of this means you can burn or reburn any location(s) anywhere without having to break a safety lock.

The first three lines of the program contain variables which have to be set each time you want to burn an EPROM. For this reason, the three lines must be typed into RAM with a fourth line to provide a call or jump to the remainder of the program. The rest of the program can be located in ROM (at say 0700).

Here's how it is done:

You will need a blank 2716.

The first stage is to transfer the MONitor program into the new EPROM. Load the following into 0800:

LD DE 1800	**800**	**11 00 18**
LD HL 0000	**803**	**21 00 00**
LD BC 06FF	**806**	**01 FF 06**
LD A(HL)	**809**	**7E**
LD (DE),A	**80A**	**12**
PUSH BC	**80B**	**C5**
DJNZ FE	**80C**	**10 FE**
DJNZ FE	**80E**	**10 FE**
POP BC	**810**	**C1**
INC HL	**811**	**23**
INC DE	**812**	**13**
DEC BC	**813**	**0B**
LD A,B	**814**	**78**
OR C	**815**	**B1**
JR NZ	**816**	**20 F1**
Restart 0000	**818**	**C7**

Make sure the switch selects 2716. Push RESET, GO. The TEC screen will blank for about 2 minutes while the program is burning.

When the screen reappears you can check the operation by addressing 1000 and read the locations. Compare them with 0000 and confirm the program has been transferred.

The next stage is to add the burn routine to the MONitor ROM. This is done at 0700. Change the values at 0800 to:

11 00 1F
21 09 08
01 10 00

Push RESET GO and the program will be transferred in a few seconds.

You have now produced a MONitor ROM with a burn routine at 0700. Place the new ROM into the TEC and it will start up with 0800.

To use the BURN ROUTINE, type the following at 0800:

11 ____ ____ TO: ROM address + 1800H
21 ____ ____ FROM: RAM address
01 ____ ____ No of hex bytes
C3 00 07

Programs to be burnt into EPROM are placed at **0900** and can extend to **0FF0.** To transfer these programs to EPROM, place the following at **0800:**

11 00 18
21 00 09
01 F0 0E
C3 00 07 Push RESET GO.

EX: 80 byte program at 0900 to 0000 in EPROM:
at 0800:
11 00 18
21 00 09
01 80 00
C3 00 07 Push: RESET, GO.

A6 byte program at 0A00 to 0180 in EPROM:
at 0800:
11 80 19
21 00 0A
01 A6 00
C3 00 07 Push: RESET, GO.

40 byte program at 0C00 to 02C0 in EPROM:
at 0800:
11 C0 0A
21 00 0C
01 40 00
C3 00 07 Push: RESET, GO.

If you type a program at **0800**, the BURN ROUTINE can be located at **0900:**
11 xx xx
21 00 08
01 xx xx
C3 00 07
decrement to **0900** Push: GO, GO.

Before starting any programming you should fill the TEC RAM with FF's. This will allow any non-program locations to be transferred and retain the value FF.

To fill RAM with FF's:

11 FF FF
D5
C3 03 08
Reset, GO.

Programs can be transferred from EPROM to the TEC memory via the following routine:
at 0C00:

11 00 08
21 00 10
01 ____ ____ (No of bytes)
ED B0
C7
decrement to **0C00 Push: GO, GO.**

Program will transfer very quickly - This is not a burn routine but a DUMP ROUTINE which can also be used for the non-volatile RAM project.

Example: 80 bytes in EPROM at 0000 to 0900 in TEC RAM.

at 0800:
11 00 09
21 00 10
01 80 00
ED B0
C7 Push Reset, Go.
(0000 in EPROM means page-zero in EPROM).

Ex: **B0 bytes in EPROM at 0630 to 0900 in TEC RAM.**

at 0800:

11 00 09
21 30 16
01 B0 00
ED B0
C7 Push: Reset, Go.

Before attempting any transfer, you must write the necessary program on a piece of paper using one of the examples in the text. Check it carefully then type it into the TEC at 0800 (or other location as explained).

Using the programmer and the non-volatile RAM in conjunction with the TEC will open up lots of possibilities. Programs can be used on the TEC or MICROCOMP and you will begin to see how everything is going together.

SHOP TALK

We're still here and have been working non-stop to bring you this issue . . . and more!

Between one issue of the magazine and the next, we go through highs and lows which would sink even the hardiest of businesses.

Our recent spate of shocks is no exception.

We have had three separate incidents, each of which contributed to the extended delay between issue 12 and now. One is so damaging that we cannot mention it at all, while the other two are not only damaging but on looking back, generated quite a laugh.

We can't say much more without treading on toes and possibly endangering future ventures, but the theft we have incurred over the past 8 months has been in the order of thousands of dollars. It comes under the heading of organised crime.

If you think you can run a business and not be touched by one or more of three forms of theft, you are sorely mistaken.

The three forms come as customer theft, staff theft and organised crime.

We uncovered varying forms of each, not only in our business, but also in consulting appointments, ranging from 5% to 15% of turnover. This equates to between 20% and 60% of profit!

I thought the electronics industry and related fields was devoid of this corruption but how naive I was. There's always someone smart enough to spot a loophole and nestle in on a cumfy innings. Some get caught, others are one step ahead of the law.

Sometimes you can count your losses with absolute accuracy but are completely powerless to arrest it. This is the frustration many businesses must endure and if you see a business with flashy surroundings and all the pizzazz of success, remember the owner may be counting his existance by the slightest of margins at the end of each week.

But on the brighter side, we knew, or hoped, the problems would gradually dissolve and have kept ourselves fully occupied with designing and developing ideas for future issues. We have now built up a collection of very interesting projects.

Some of these are extensions to our computer projects and others are answers to requests from readers.

It's only when we get more than three similar requests that we consider the idea worthwhile. Lots of readers have their own ideas as to what they want and it generally relates to a specific need.

Ideas have to be of general interest and simple enough to be fool-proof, before they will be considered for presentation.

In this issue we have taken up where we left off with the TEC project and increased its capability to programming EPROMs.

This will enable you to burn EPROMS for use in such projects as dedicated Micro's or for the Microcomp.

If you are wondering about the other books and publications we are bringing out at various intervals, the news is not good.

We are not going to produce any more 'one-shots' until the distribution situation is cleared up to our satisfaction. At the moment we are losing too many copies to an ineffient distribution system and not recouping even our printing costs.

We don't know the true situation but as far as the monetary return is concerned, some of the one-offs have been very successful. Others have been a total disaster, leaving us with a overall situation where we barely broke even. This is not satisfactory when you consider the thousands of hours which go into the production of a book.

One the success side is Stage-1. It proved so successful that we reprinted it about a year ago and have now run out of both releases.

Due to the high cost of reprinting, we have decided not to issue it again as a book but rather release it in instalments in TE magazine.

This will be welcomed by those who were going to send in for a copy.

Some schools have used Stage-1 for the past 3 years and its seems a pity to have run out just when it is getting popular.
But unfortunately our low cover prices do not allow for small reprints and it's one of the anomalies we have to face.

Digital Electronics REVEALED will be the next to go out of print as stocks are down to the last 300 copies. If you are currently doing a course in digital, this book will prove invaluable. The price is still $2.90 per copy or $2.30 for class sets, plus postage.

Talking about postage, this perennial increase has come upon us once again. At one time, postage was such a small part of the cost of sending a product that it was not included in the price. In fact it was an insult to say "plus postage".

But gradualy this cost has crept up and up to a level where it represents a very high proportion of the costs.

In fact it is pricing itself out of the market, so much so that electronic transfer of data, documents and monies is taking over.

With the near introduction of ZIP code addressing, you will soon be able to identify the boundaries of the locality you are sending to.

Extending this further, you will be able to 'key' the ZIP number into a memory bank near the post box and the letter will be 'tagged' from the moment it is posted.

The stored code will accompany the letter throughout its entire journey and offer many facilities not currently available.

You could request it be withdrawn, re-directed, or merely check that it has reached its destination.

These possibilities are almost with us and within a few years we will begin to see them come into being.

All this capability depends on microprocessor systems and the only way to keep up and/or understand how they operate is to build your own system and teach yourself.

The micro got an enormous impulse via the Personal Computer market. In 5 years it advanced from slow-moving, chunky graphics to full colour, high resolution, fast moving, priority coded, text-book graphics.

Even though the advances were enormous, the satisfaction to an electronics buff was minimal.

To get into the workings of any of these computers was almost impossible and expecting to be able to modify or interface to the computer was wrought with disaster.

Even the simplest of requirements such as outputting to relays or LEDs was hampered with lack of techincal data.

Manufacturers expected the computer to be used 'as is' and the only real back-up facility was an assortment of games. You could really say computers were merely a complex games machine. Instead of going to the pin-ball parlour, the personal computer brought the video games into the home.

Without being too denigrating, I think I can say with surety that their policy was an enormous success but had a very short life. The PC market and video games market has now died; and died completely. The whole market is dormant with nothing being bought and very little being played.

We are yet to see the next great money grabber but the quality and capability of PC's will have to increase appreciably for them to attract the market again.

That's why we have steered clear of flashy PC market and concentrated on the learning side of the micro.

For us, the TEC started a revolution. Not only did it show and teach both you and us the capability of micro designs but it enabled us to design an even simpler computer.

THE MICROCOMP

After a lot of designing and experimentation we have come up with world's simplest computer, using readily available, inexpensive, components. It uses just 3 chips and a handful of small parts to create a dedicated system which can easily be expanded to cater for all sorts of requirements.

We have already produced over 15 programs for it and added RAM memory, additional displays, relays and input devices to create effects which would normally take many, many chips in a conventional design.

You should also consider the advantages of a micro design viz:

When you find the project does not work to your complete satisfaction, instead of having to redesign the circuit, all you have to do is rewrite the program and the modification is done.

Surprisingly, a micro-design is cheaper to implement than a project using discrete components and we can confidently say that it is more reliable and easier to get working.

Finally, and most important, the cost of a micro system is less than a conventional deisgn if you consider ideas as 'add-ons' to a base design. For a few dollars you can create add-ons for specific applications and with a range of plug-in modules like this, you get enormous flexibility.

We have spent the past two months concentrating entirely on getting the microcomp working and designing a range of plug-in modules.

This issues sees the beginning of the article and shows how to construct the computer and get it going. The EPROM which comes with the kit is filled with a number of programs and the first of these appears in the article.

Our latest introduction into microcomputer understanding

THE MICROCOMP.

Programming is a new facet of electronics and its power cannot be over-emphasised.
After all, programming is the only thing limiting you from launching into industrial designs.

Computer circuits have been around for many years and even a personal computer can be adapted to your own specific needs, with a little bit of electronic skill.

But it is the programming which is the key to getting an idea operational.

This is where we come in. The last 15 or 16 pages of this issue introduce the fundamentals of programming and at a later stage we will include some more complex skills.

With these as building blocks you will generate a knowledge of programming and be able to create programs of your own and match them perfectly to the surroundings. This way they become tailor-made and as you gain confidence, you can interface more and more devices to the system.

Read our Microcomp project and see how it fits into the microprocessor field. It is intentionally aimed at those who have never built a micro system before and feel 'it is not for me!'

Unless you get hands-on experience in this area, you will be left out in the cold. Micro systems are by far the best and simplest way to go.

Once you get involved, a whole new world of possibilities will open up. All those ideas you have harboured for years will suddenly become possibilities. Knowledge on processors and programming will keep you ahead for the next few years and I guarantee you will be appreciated by your employer.

If the TEC passed you by, don't let the Microcomp follow. The two are not the same and we have both in our work-room. You will find each has different applications and you will learn differently from each.

It does not matter which you start with. They will both let you graduate.

As I said, we spent two months enjoying ourselves, writing programs, designing circuits and producing circuit boards. The challenge was great and on more than three occassions we encountered a bug which took up to 8 hours to solve!

Fortunately you won't have these problems because the circuits are now in PC board form. But many of the ideas we had to design from scratch and had very little back-up information to assist us.

We chose the long, slow hand-assembled way to create a program to prove that almost anything can be produced simply by sitting down with pen, paper and a set of MACHINE CODES.

Putting your thoughts into a form which can be understood and executed by the micro is the major feature of this project and we have taken great pains to explain each and every line in the programs.

I don't think there can be any genuine excuse for not launching into micro designs. If you want to stay in this field, it is essential.

Microprocessor designs are appearing at every level of electronics, from the hand-held game to satellite navigation.

We have done our utmost to make the project attractive, economical and interesting. You'll only be kidding

yourself if you think you can stay in the field while skirting any designs starting with 'M'.

FOR THE BEGINNER

Don't worry, we havn't forgotten the middle-level constructor. We always include projects for those starting out and have included a simple project in this issue.

Some junior constructors have a tendency to want to make their own PC boards.

While this is ok for one or two projects, don't get its value out of perspective.

Making circuit boards is a very small part of learning electronics and if you really knew the truth, you would be surprised. We consider the manufacture of a PC board has nothing to do with electronics and you will find very few design laboratories make their own boards.

They send the artwork to firms specifically set up to make 'one-off' prototypes.

This may sound expensive but a single board will cost about $20 and this works out cheaper than drilling and etching the board yourself. So don't spend too much time on making PC boards but rather study the circuit and how to make a neat component layout.

If you are contemplating making a board, it is best to spend a little more time and create the artwork with tape and stick-down pads. Even if you want to make only a single copy, this method makes a far superior pattern.

It is called the PHOTOGRAPHIC method and means you can make multiple copies, if needed.

In any case, making your own board costs more than buying the ready-made equivalent and the dangers of the chemicals are only just recently being realized. I don't want to spoil the fun but put the emphasis where emphasis is due.

STAGE-1

We mentioned the complete sell-out of ELECTRONICS Stage-1. To assist those who have missed out, we have decided to reprint it near the middle of the magazine so that it can be pulled out and stapled to form a book. The first section appears in this issue.

FM BUG

The most common fault we have found with the FM bug is the diameter of the aerial coil.
It must be wound on a 3.5mm screw-driver and if you have a 3.5mm phono plug, you can wind it on this!

Corrections for issue 12:

P 20. Col 1. Twelve lines from bottom: **18 02** Sixteen lines from bottom: For a forward jump **02** will ....
P. 30: 1 - 4069 IC in parts list
P. 58: 1 - 4069 for Printer Interface.
P.60: 2- BC 547 transistors for Touch Puzzle.
P.72: 2 - BC 547 transistors for Parts List.

Individual parts for any of the projects are available from TE. We don't normally sell separate items but are prepared to help you out if you get stuck. Write or ring and we will let you know the cost and availability. All kits use commonly available parts and you shouldn't have any trouble.

STEREO AM

Stereo AM is the best thing to happen to AM for 50 years!!

AN **AM UPDATE** letter arrived in the office today and aroused discussion between the staff. It's not very often they voice their thoughts!

The introduction of stereo transmission for the AM band will give radio its biggest impact for years. Previously it was the introduction and expansion of the FM band into stereo and the stations hopped on the advertising bus with NOISE FREE RECEPTION and STEREO TRANSMISSION!
Now AM is going stereo and if the way the market has provided for the introduction is any guide, it looks like it will be quite a success.

Although FM is undoubtedly clearer and freer from interference, it is not suited to portable receivers in moving situations.

It seems FM is prone to drop-out in hilly locations and has a tendency to fade when in a car etc.

Hopefully AM will be the answer everone is waiting for. It is already the favourite of 8 out of 10 listeners and this is mainly due to the large number of stations offering a wide range of listening material.

The designers of the system claim the output of the new stereo sets is superior to older AM sets and this will produce better frequency response in the output.

This will be of special interest to motorists who opted for the clearer, crisper sound of FM to complement the cars appointments.

With improved AM the worry of fading or limited range will be a thing of the past.

The only point to be careful of when buying an AM stereo receiver is the wording on the carton.

You must make sure it specifically states AM STEREO and not AM/FM stereo.

You should also ask for a demonstration before buying as many shops will be trying to get rid of their old mono stock.

I can't ever see TE going into AM/FM kits as the number of components is high and many of them are not available on the hobby market. A kit would possibly cost more than the retail equivalent.

Because AM stereo products have only just been released, their prices are still fairly high. If you can't wait, here is a list of what is available:

★ AWA has released the Clarion Stereo AM car radio/cassette player model 990E. It retails for $550.

★ Concord Auto Hi-Fi has a new Stereo AM car radio/cassette player, the HPL 550.

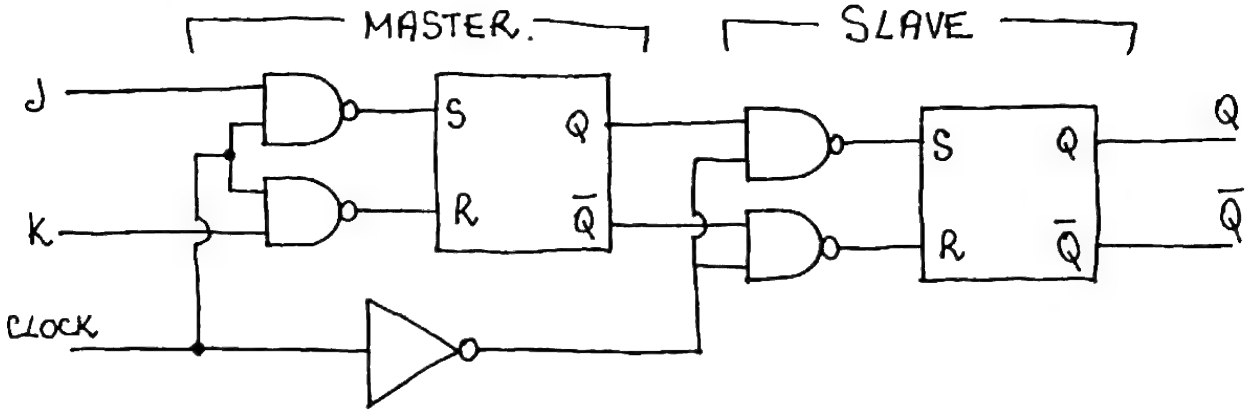
★ Eurovox have a stereo AM car system equipped with Dolby sound. They are called 2301 and 2300 and will cost around $700.

★ Pioneer Electronics have a stereo AM/FM radio/cassette unit, the KE-433AM. It will retail for $350 and is compatible with existing booster amplifiers and speakers.

★ Sansui produces some upmarket models including the TUS 77 AMX tuner for $739, the SX 1070, a 55 watt per channel receiver for $699, the SX 1100, 100 watt per channel receiver with both AM/FM capacity for $1399 and the CX 990 car stereo with auto-reverse cassette radio for $689.

★ Sony have two models. One is the SRF-A200, a slim portable radio with built-in speakers, retailing for $119 and also the SRF-A10, a Walkman receiver (without cassette) for $99.

THE JK FLIP FLOP CONSISTS OF 2 SIMPLE R-S FLIP FLOPS AND GATING CIRCUITRY



BASIC J-K FLIP FLOP.

THE INPUT LINES CONTROL THE MASTER LATCH. THE MASTER FEEDS THE SLAVE & THE OUTPUT IS TAKEN FROM THE SLAVE.
THE CLOCK LINE IS DESIGNED TO CONTROL BOTH SECTIONS & THE INVERTER IS POSITIONED BETWEEN THE TWO TO ALLOW ONLY ONE TO BE OPERATIONAL AT A TIME. NOTE: DIFFERENT TYPES OF FLIP FLOPS ARE MANUFACTURED, SOME HAVE ACTIVE HIGH CLOCK LINES & OTHERS ARE ACTIVE LOW.
OUR DISCUSSION USES AN ACTIVE HIGH CLOCK LINE TO OPEN THE MASTER SECTION. THIS WILL ALLOW THE INFORMATION ON THE J&K LINES TO SET OR RESET THE MASTER LATCH.

AT THE SAME TIME THE HIGH CLOCK LINE IS INVERTED & APPEARS AT THE SLAVE SECTION GATING WHERE IT PREVENTS ANY SIGNALS ENTERING THE SLAVE LATCH.

WHEN THE CLOCK LINE GOES LOW, THE DATA ON THE J & K LINES IS FROZEN INTO THE MASTER LATCH & THE GATES BETWEEN THE TWO SECTIONS OPEN TO ALLOW THIS INFORMATION TO PASS INTO THE SLAVE LATCH. THE RESULT IS AVAILABLE ON THE OUTPUT(S).

BUT JUST A MOMENT....

CAN THE STATE OF THE JK LINES BE ANY COMBINATION OF HIGHS & LOWS? TO ANSWER THIS WE WILL HAVE TO LOOK AT THE CONTROL GATES. THESE ARE NAND GATES & JUST TAKING THE J LINE, THE TRUTH TABLE SHOWS THE J INPUT MUST BE HIGH FOR THE CLOCK TO HAVE ANY EFFECT. THIS MEANS THE J INPUT CAN ONLY BE HIGH TO HAVE ANY EFFECT.

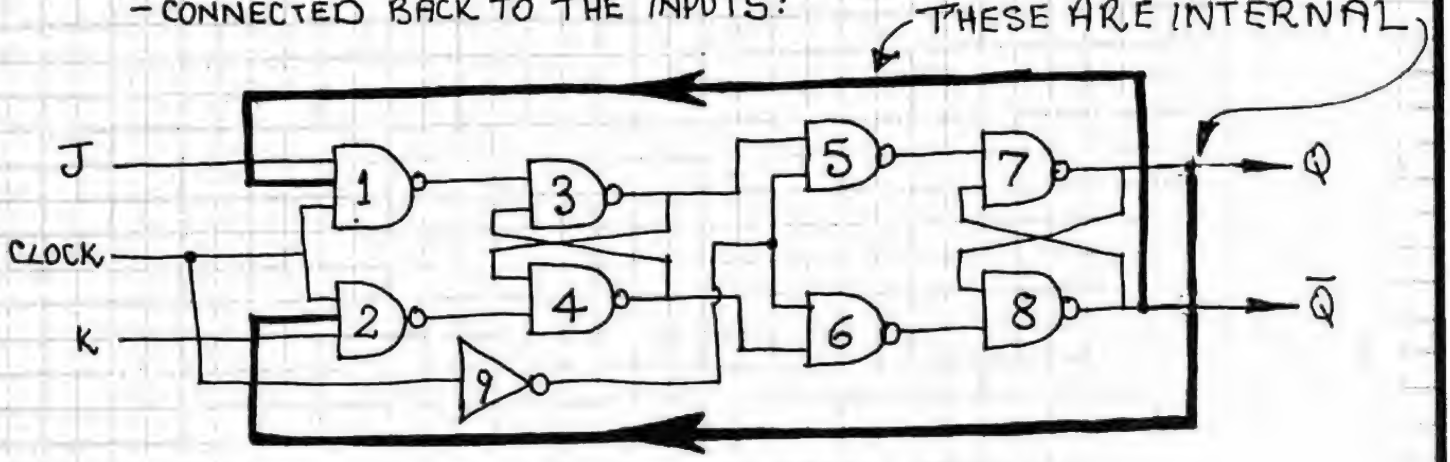
NAND

J INPUT	CLOCK	OUTPUT TO THE MASTER LATCH
0	0	1
0	1	1
1	0	1
1	1	0

OUR PREVIOUS STATEMENT SHOULD BE MODIFIED TO "THE HIGH DATA ON THE J&K LINES WILL BE PASSED TO THE MASTER LATCH".

WHAT WILL OCCUR IF BOTH THE J&K INPUTS ARE HIGH?
THE INPUTS TO THE MASTER WILL BOTH BE LOW & THIS WILL BE PASSED TO THE SLAVE LATCH TO PRODUCE AN UNDESIRABLE OUTPUT.
TO PREVENT THIS UNWANTED CONDITION, THE OUTPUTS ARE CROSS—

- CONNECTED BACK TO THE INPUTS:



JK FLIP FLOP WITH CROSS-COUPLING.

THIS CROSS-COUPLING EFFECTIVELY PREVENTS BOTH INPUTS BECOMING HIGH AT THE ONE TIME. — ONE MUST BE LOW.
THE MAIN FEATURE OF GATES 1&2 IS: ALL 3 INPUTS MUST BE HIGH FOR THEM TO CHANGE. THIS IS NOT POSSIBLE WITH CROSS-COUPLING AS ONE LINE SHOWN IN THE ABOVE DIAGRAM IS ALWAYS LOW.
LET US SEE HOW THIS AFFECTS THE OUTPUT:

HOW A J-K FLIP FLOP PRODUCES A HIGH ON THE OUTPUT

WHEN POWER IS APPLIED TO THE CIRCUIT THE NATURAL TENDENCY IS FOR BOTH LATCHES TO SETTLE IN ONE STATE OR THE OTHER. THE MOST IMPORTANT IS THE SLAVE. ASSUME IT SETTLES IN THE RESET STATE. THIS MEANS THE Q OUTPUT WILL BE LOW AND THE $\bar{Q}$ WILL BE HIGH.

THE $\bar{Q}$ OUTPUT IS FED AROUND TO [illegible] INPUT SO THAT WHEN A HIGH IS PRESENT ON THE J LINE, GATE 1 WILL PASS A LOW TO THE MASTER LATCH & GATE 2 WILL PASS A HIGH. THE MASTER LATCH WILL BE SET. THE CLOCK LINE MUST BE HIGH FOR THIS TO OCCUR.

WHEN THE CLOCK LINE GOES LOW THE OUTPUT OF THE INVERTER WILL BE HIGH SO THAT THE CONTENTS OF THE MASTER LATCH WILL BE PASSED TO THE SLAVE. HERE THE HIGH ON THE NORMAL OUTPUT OF THE MASTER COMBINES WITH THE CLOCK LINE TO PRODUCE A LOW OUT OF GATE 5 & A HIGH OUT OF GATE 6. (VIA THE LOW FROM GATE 4.)
THESE SIGNALS "SET" THE SLAVE LATCH AND THE NORMAL OUTPUT Q CHANGES TO A HIGH.

IF J GOES LOW THE LATCH WILL NOT CHANGE STATE, EVEN IF THE CLOCK IS APPLIED A NUMBER OF TIMES.

THIS IS DUE TO THE 3-INPUT AND GATE NOT CHANGING ITS OUTPUTS TO THE MASTER LATCH.

THE ONLY WAY TO CHANGE THE FLIP FLOP IS VIA THE K INPUT & A CLOCK CYCLE..... THIS IS NEXT.....

HOW THE J-K FLIP FLOP PRODUCES A LOW ON THE OUTPUT

WE WILL ASSUME THE FLIP FLOP IS IN THE HIGH CONDITION FROM THE PREVIOUS EXAMPLE. THE J LINE & THE CLOCK CAN DO NOTHING TO CHANGE THE FLIP FLOP FROM THE SET CONDITION.... IT NEEDS THE K LINE.

WHEN A HIGH IS APPLIED TO THE K LINE (THE J LINE SHOULD BE LOW FOR THIS OPERATION) GATE 2 WILL RECEIVE A HIGH FROM THE FEEDBACK LINE (COMING FROM THE Q OUTPUT) SO THAT A HIGH FROM THE CLOCK WILL CHANGE THE OUTPUT OF GATE 2 TO A LOW, & THE MASTER FLIP FLOP WILL BE RESET.

WHEN THE CLOCK CHANGES TO A LOW THE OUTPUT OF THE INVERTER WILL BE HIGH AND ALLOW THE RESET CONDITION TO RESET THE SLAVE LATCH & THUS THE Q OUTPUT WILL GO LOW.

IF THE K INPUT IS KEPT HIGH & CLOCK PULSES ARE APPLIED (TO THE INPUT) THE FLIP FLOP DOES NOT CHANGE STATES — IT REMAINS LOW.

TOGGLING

WHEN THE J-K LINES ARE BOTH HIGH AN AMAZING THING HAPPENS. THE FLIP FLOP TOGGLES.
THIS IS WHAT HAPPENS....

TOGGLING MEANS THE OUTPUTS CHANGE (WE ARE REFERRING TO THE NORMAL OUTPUT Q) FROM HIGH-TO-LOW-TO-HIGH-TO-LOW etc WHEN A CERTAIN SET OF CONDITIONS IS APPLIED TO THE INPUT LINES 1 & 2 AND THE CLOCK LINE GOES HIGH-LOW-HIGH-LOW etc. THE CONDITIONS ARE THESE:

THE J & K LINES MUST BE KEPT HIGH & THE CLOCK CHANGES FROM HIGH-TO-LOW etc.

BUT THAT'S NOT QUITE ALL. THE RELATIONSHIP BETWEEN THE NUMBER OF CYCLES REQUIRED BY THE CLOCK TO PRODUCE ONE CYCLE AT THE OUTPUT IS A MOST INTERESTING OBSERVATION.

L THE FLIP FLOP IS INITIALLY IN THE RESET MODE AND THE Q OUTPUT
L IS LOW. AND THE CLOCK LINE IS INITIALLY LOW.
H WHEN THE CLOCK LINE GOES HIGH, GATE 1 HAS THREE HIGHS ON THE INPUTS & THIS SETS THE MASTER LATCH.

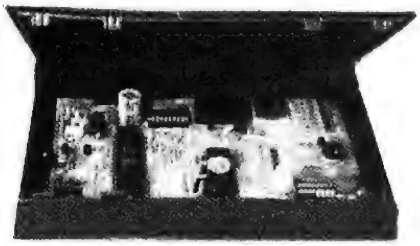
L WHEN THE CLOCK LINE GOES LOW THIS "SET" CONDITION IS
L PASSED TO THE SLAVE LATCH & THE OUTPUT IS CHANGED TO A
H HIGH. WHEN THE CLOCK LINE GOES HIGH, GATE 2 RECEIVES THREE HIGHS ON ITS INPUT LINES (OF WHICH ONE IS THE FEEDBACK LINE FROM THE Q OUTPUT) & THE MASTER LATCH IS RESET.

L THE CLOCK LINE GOES LOW AND THE CONTENTS OF THE MASTER
L LATCH IS PASSED TO THE SLAVE AND IT IS RESET. THE NORMAL
L OUTPUT CHANGES TO A LOW.
↑ CLOCK CHANGES.

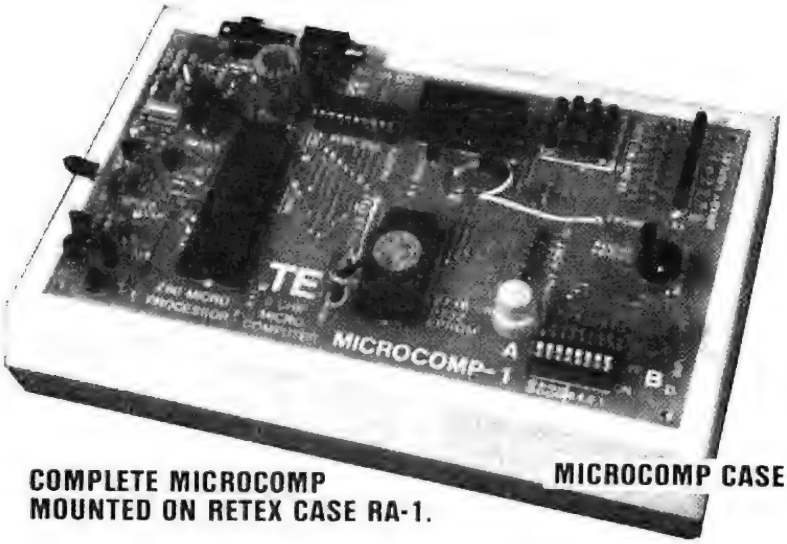
MICRO COMP

A 3-CHIP Z-80 COMPUTER

$55.75 COMPLETE
COMES WITH **FREE** STORAGE BOX!!



Parts: $47.25
Board: $8.50



COMPLETE MICROCOMP MOUNTED ON RETEX CASE RA-1.

MICROCOMP CASE $12.50

Don't think this project displaces the TEC. It goes hand in hand with it and uses some of the 'add-ons' and capabilities to assist in preparing programs. In reality you need BOTH. The TEC produces the programs and the MICRO-COMP runs them.

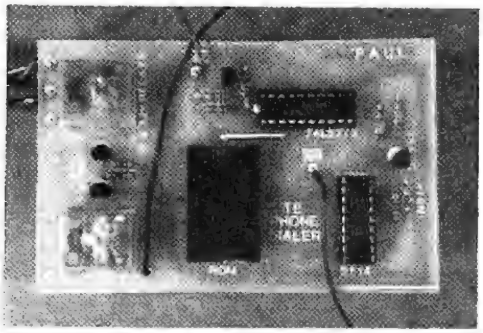
But if you are only starting in this field and want a low-cost introduction into micro-computer programming - THIS IS IT!

There are some pre-requisites however. Although the project is simple (according to computer standards), it needs a degree of competence in assembly and you should have constructed at least 6 other TE projects before attempting it. Digital projects have an inherently high degree of success however construction requires a fine tipped soldering iron and a lot of attention to detail.

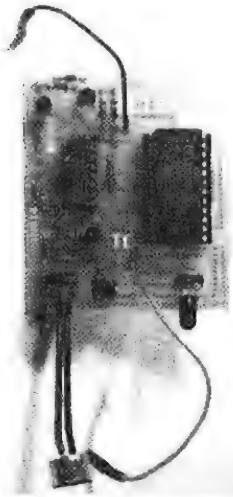
If you are not up to it, don't do it. But if you have made a lot of projects and want to graduate to the next level - this is how to go. You will be amazed with the capabilities of the unit and it will involve you in hundreds of hours of programming.

At the conclusion of this project you will:
- Know a little (a lot) about the Z 80 microprocessor.
- Learn about Read Only Memories and Random Access Memories.
- Learn about Input and Output ports and devices.
- Learn how a Micro system works
- Learn how to produce MACHINE CODE programs.

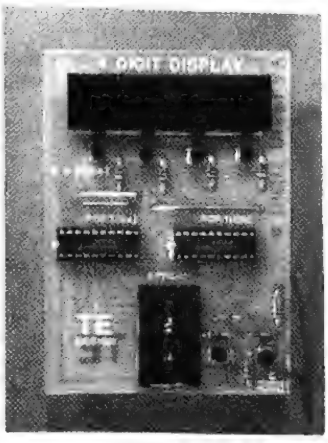
SOME OF OUR 'ADD-ONS':



PHONE DIALLER



MORSE TRAINER



4 DIGIT DISPLAY

This project contains everything to get you started in Machine Code programming. It assumes you know nothing about micro-processors or how they operate. The MICRO-COMP is the simplest computer to be offered on the market. It uses only 3 chips and a handful of small components to prove that computers can be tackled and mastered by anyone interested in electronics.

As with all our projects, full kits are available and come with a complete back up service.

As we have said, it doesn't displace the TEC but complements it. And yet the MICROCOMP is a stand-alone project. It is self-contained and comes with a range of interesting programs as shown on P. 70.

Ten programs have already been designed and come in the pre-programmed ROM. They include COUNTING, DISPLAYING and GAMES. The readout for these is via the displays on the board but as the games become more complex we have designed plug-in modules which connect to the main board via a wire-wrap/component header plug similar to the arrangement used in the TEC.

This has proven to be the neatest and most rugged way of adding features and allows you to increase and extend the capability of the system to quite high levels.

Some of the plug-in boards run two or three programs and by building all the modules, you will be able to run all the programs in the EPROM.

Programs which have already been completed include:
- MORSE CODE TRAINER
- MORSE RECEIVER
- TELEPHONE DIALER
- COUNTING
- MASTERMIND
- JUMP RELATIVE routine for determining the value of the displacement byte.

The MICRO-COMP can also be combined with the TEC and they go together perfectly. With the assistance of the TEC you can create your own programs, burn them via the EPROM BURNER or hold them in our non-volatile RAM card for running on the Micro-comp. This non volatile RAM project is equivalent to TAPE-SAVE and has the advantage of being able to be transferred instantly or run as a ROM.

It is a battery backed-up 6116 and when in the power-down mode, consumes less than 2 microamps. Two AAA cells power the unit and are capable of holding the information for about a year.

The TEC and MICRO-COMP provide a complete designing system for creating Machine Code programs and you can use the Micro-comp for the execution of the program.

The Micro-comp comes complete with a 2732 EPROM which is filled with lots of programs. All these have been produced entirely on the TEC and tested on the Micro-comp. We have not had the assistance of a compiler, video display or Z-80 simulator, proving that programs can be generated 'by hand'.

Agreed, this means it has taken longer to create the programs but the challenge was well worth it. Even the concept of a half-byte memory for the Phone Dialler was an innovation never before tried.

We admit Machine Code is not a fast method of creating programs but has the advantage that almost anything can be turned into a program. And it can be done with the simplest of equipment. The only limitation is the programmer's skill.

By building both the TEC and Micro-comp, you gain first-hand knowledge of two different methods of designing a micro system. You will also have need for add ons such as the non-volatile RAM and EPROM burner.

In effect you will be a self-contained programming station capable of turning out 'one-offs' or mass copying your own programs.

Please remember: These notes use simple terms and simple explanations to make programming easy. Although they are accurate, they do not cover everything and we suggest you purchase a couple of books on the Z-80. The two best books to buy will be given later.

Since its introduction, the word MICRO has been the most feared in the industry because, up to now, the operation of a microprocessor system has been very much a mystery.

Never has a writer explained or presented a system which could be understood by beginners. They argued it wasn't for beginners but everyone must be a beginner at some time. Because of this, Micro's have been a closed topic to the newcomer and this amazing electronic device has been left for the clever ones.

Now this has all changed.

The MICRO-COMP is here. With only 3 chips it is even simpler than a medium sized 'regular' project and yet its capabilities are beyond belief.

For the 'brains' of the unit we have used the Z-80. The most popular microprocessor on the market. Why the most popular? Because, up to now, industry swallowed them up totally and consumed the entire production. It's only with the slump in computer and games sales that supplies have reached the hobby market. And due to manufacturing efficiency, these truly amazing chips have come down to only a few dollars.

This means the project will be well within your budget.

Apart from the programs already mentioned, we are in the process of producing programs and modules for an Alcohol Breath Tester, a Digital Resistance Meter, Digital Capacitance Meter, a Bio-feedback Unit, a Mini Frequency Counter and a Lung Capacity Meter.

But before we get too carried away, let's look at the project in detail:

THE MICRO-COMP.

This is a 3-chip computer capable of accepting input data, performing operations on this data and displaying the results. The amazing part of this project is the three chip count. To achieve this we have used some very cunning circuit designs, some of which cannot be translated to larger designs. However our aim has been to produce a computer which will execute Z80 Machine Code with the least number of chips. And this we have done.

The reason for the minimum chip count is simple. Most constructors count the chips in a project before starting and anything over six scares nearly everyone away. With 3 chips, ma ıy will 'give it a go' and that's where we win. We want lots of readers to try their hand at construction and experience the excitement it offers.

Everyone has seen micro systems in a hard-to-get-at form. The **Personal Computer**. But these have never enabled you to get into the 'works' or let you find out how they operate. You only get to see the end result -the print-out or Video picture.

The Micro-comp is designed to break this barrier. With only 3 chips you will be able to follow a 'minimum parts' system and understand what is going on. Even with 3 chips you can

use nearly all the Z 80 commands and create an endless number of programs.

This project is really a software project. Building the Micro-comp is purely secondary. But how can you learn about programming without experimenting with the real thing? So building the Microcomp is really an essential part of learning to program.

Its price is low enough for everyone to afford and it has an end-use around the home as a controller for lighting or security which would match any commercial unit. You could also use it in your hobby, model railway layout or as a timer-controller in industry.

The MICRO-COMP doesn't do much when compared with a Personal Computer. But that's not its purpose. It is intended to teach Machine Code programming, the code behind all computer instructions.

The only instructions a processor understands is Machine Code. All other high level languages have been invented to allow humans to understand what is going on. Languages such as BASIC and FORTH provide a connecting link between the micro and the human mind. This means all inventors of languages have had to use machine code to write their programs. So why not use Machine Code direct?

Using BASIC is like hiring a scribe. Centuries ago people could not write. So they went to a learned man and told him what they wanted written. After a lengthy discussion he would write a letter. The letter represents Machine Code. The lengthy discussion represents BASIC.

BASIC has its advantages and a number of disadvantage. Its advantage is it gets you into a microprocessor system with very little effort and undertanding. But its disadvantage is it needs the 'scribe' to be present at all times. With machine code its like using the typewriter yourself.

But most important Machine Code is the best way for producing programs for controlling applications. When you consider all video games and industrial machines are Machine Code based, you will see where the future lies.

It is interestng to note that a micro system rivals a 'normal' project (using individual chips) when as few as 10 chips are involved. When you consider a microsystem can be modified and altered to suit changing circumstances, it is clearly the only way to go.

Why this hasn't been the case, is simply due to fear.

Everybody thought microprocessors were complex mysteries and preferred to stay with the building blocks they knew and trusted.

But, in fact, the micro system is simpler. Once the basic design is built, it only requires programming to perform the required function. To change the function, the electronics don't need altering, only the program!

Micro systems are simply thousands upon thousands of building blocks stored in the form of program and to write a program is equivalent to being able to create your own chips.

This is what the MICRO-COMP is. You can get it to execute your own programs and connect all sorts of input-output devices. You can get it do just about anything in the controlling and timing field but first you have to learn how to program.

To help you with this we have produced a number of programs to demonstrate the capabilities of the system and these are contained in the lower half of a 2732 EPROM which comes with the kit. Later you will be able to send it in for re-burning for the additional programs.

Before we get into the construction of the 'Comp, here's a brief discussion on how it works.

HOW A COMPUTER WORKS

This is a very simple explanation to get you started.

The operation of a computer revolves around a chip called the CPU. This applies to any computer and the MICRO-COMP is a computer, even though it is very simple. In our case the CPU is a Z 80. It is the 'brains' or 'clever chip' in the system and controls all the other chips.

CPU stands for Central Processing Unit and the feature which makes it so clever is it is good at organising things. It keeps the whole system operating and running smoothly.

In an audio or radio circuit there is usually only one signal path. In a computer there are lots of signal paths. This is the one striking difference between the two. In a radio, the path can be tapped at any point and you will be able to hear the signal (such as voice or music). If you tap any of the paths in a computer you will hear a series of clicks or tones and they will not make any sense.

This is because a computer requires a number of lines carrying signals AT THE SAME TIME to produce the necessary commands and output effects.

A single line in a computer will sound like a tone because of the high speed of operation but as far as the computer is concerned, the line is producing a HIGH for a very short period of time and then a LOW for the remainder of the time.

Since a single line can only produce a HIGH or LOW, a group of lines is required for the transmission of numbers. This is achieved by assigning the lowest-value line with '1', the next line with '2'. the next with '4', the next with '8', the next with '16' and so on.

By turning on combinations of these lines, almost any value can be transmitted.

A group of lines such as this is called a BUS and a computer has two buses. One consists of 8 lines and is called the DATA BUS, while the other has 12 or more lines and is called the ADDRESS BUS.

The microcomputer starts operating after the reset button has been pressed and released. This action resets the Program Counter inside the Z80 to 0000 and instructs the chip to fetch 8 bits of information from MEMORY.

It does this by putting zero's on all the address lines and turning on the 2732 via the Chip Enable line.

The EPROM responds by delivering the 8 bits of data which are located at 0000 to the data bus. The Z 80 accepts these and places them in a special instruction register which is only accessable to the Z80.

Eight bits of information is called a BYTE and the Z 80 determines what to do with the byte , according to its value.

The Z 80 will do one of two things. It will either carry out the instruction or request another byte. An instruction may consists of one, two, three or four bytes. and the Z 80 waits for a command to be completed before executing it.

Looking at the machine codes on the back of issues 11 and 12 you will notice some of them consists of one byte while other are 2, 3 or 4 bytes long. The Z 80 knows exactly how long each instruction is and knows that some contain a data byte or

displacement byte. This knowledge is inbuilt into the Z80 and only needs to be fed a simple program for it to respond.

The first byte from memory is always interpreted as an instruction and the byte or bytes which follow make up the first command. If you add a byte or delete one, at any time in a program, it will not be interpreted correctly and the Z-80 will carry out totally incorrect commands.

The Z 80 reads a program one byte at a time. It does not look ahead and cannot correct any mistakes. That's why it is important to check a program before offering it to be run.

Information passes out of the Z 80 via the address bus and into it via the data bus. After the Z 80 has processed the data, it will send the result out via the data bus. This means information moves in TWO directions along the data bus, although not at the same time.

In our case, information from the Z 80 is passed to an output latch. This latch is a device which fits between the computer and say a LED, motor or relay. The need for this chip is very important, as you will see. Data could be sent directly to a LED without using a latch and it would work. But the computer would have to stop functioning for the whole time when the LED is to be lit. This is obviously not a solution and so a chip is placed between the two which holds the 'turn-on' pulse for as long as the LED is required to be activated.

This chip is called a latch . It is merely a set of flip flops which hold the bits of information for as long as is required. This enables the Z 80 to get on with its other operations such as turning on a motor via another latch. Output devices such as LEDs and motors cannot be connected directly to any of the data lines for two reasons:
Firstly the current available in these lines will not be sufficient to operate them and secondly, the lines must be available for other purposes.

This means any device wishing to be placed on the data bus must be separated from it until the exact instant when it is required.

This is what an input latch does.

When these chips are not being activated, they place no load on the bus and allow the lines to rise up and down. This feature is called TRI-STATE as they are capable of producing a HIGH or LOW when required.

This is the basis of how a computer starts up. More aspects will be discussed later.

BEFORE YOU BEGIN CONSTRUCTION:

It is posssible to construct the Micro-comp using your own components and on your own PC board.

That's because all the parts are standard and the circuit board is fairly easy to reproduce. The 2732 EPROM can be programmed via an EPROM programmer and everything will operate perfectly.

There are only two hitches to you doing this.

First is the guarantee.
If you make the project from your own parts, it cannot be sent to TE for repair. We guarantee to fix any model made from one of our kits as we have had lots of experience at this. Mainly poor soldering joints, jumper links cut before soldering, parts inserted the wrong way around and broken tracks. Small faults but enough to keep the project from working.

Digital electronics is extremely reliable but not if you make a mistake.

The second hitch is reliability. If you use second-hand or unknown components, how do you know if they are perfect? They may have been over-worked or damaged in a previous project and fail when put in the Micro-comp.

Making your own Board?

The PC board is not as easy to make as it looks. One mistake in its etching and a track may be etched through. Or a hairline crack may be created in one of the lines which will be extremely difficult to spot. You also have to consider the overlay and solder mask. These make the project look neat and professional. You may save a few dollars at the start but end up costing more in the end. We have had a few troubles with home-made boards and unless you have made lots of boards before, we suggest buying a ready-made board.

Building from a kit is the safest way. All parts are absolutely brand-new and chips are transferred from bulk tubes without being handled. Boards are inspected three times during manufacture and made on semi-automatic equipment with very little margin for error. A sample kit is constructed before they are released and at least three prototypes have been made before the project goes to print.

This contributes to the success of our kits and the neatness of the finished project is enhanced by the solder mask on the underside of the board. This prevents solder sticking to unwanted areas and shorting between tracks.

To be sure of success, buy a kit. A number of shops are selling these kits and you will find the cost is less than hunting for the individual bits yourself.

PARTS LIST

1 - 10R
8 - 100R
1 - 330R
1 - 470R
1 - 3k3
1 - 4k7
14 - 10k
3 - 22k
2 - 39k
4 - 100k

1 - 100k mini trim pot

2 - 1n green cap
1 - 100n
1 - 1mfd 63v electro
1 - 1,000mfd 25v electro

9 - 1N 4148
4 - 1N 4002

1 - 5mm red LED (SPEED)
1 - 5mm green LED
24 - 3mm red LEDs
8 - BC 547 transistors
1 - BC 557 transistor
2 - FND 560

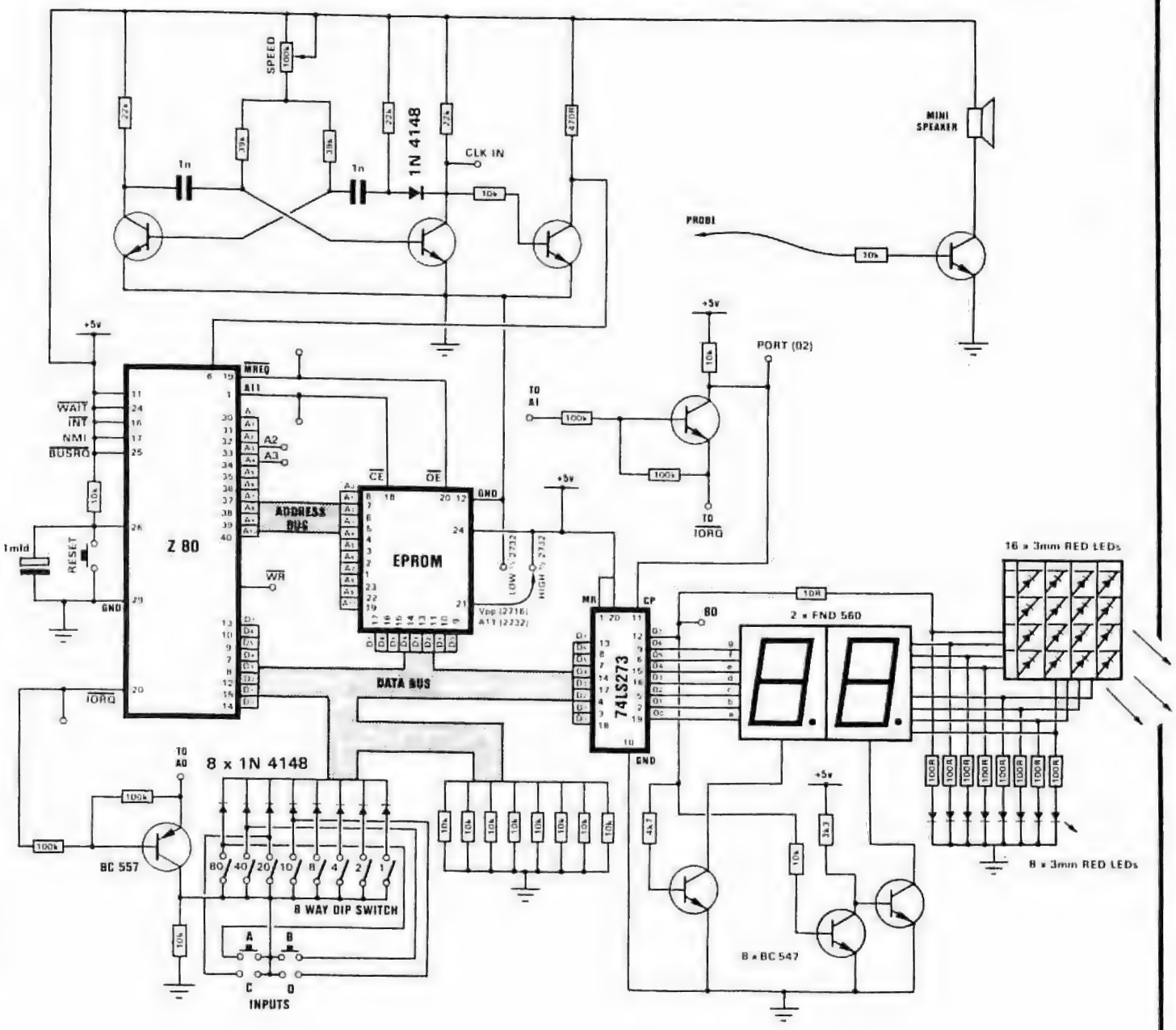
1 - 74LS273 IC
1 - Z-80 CPU
1 - 2732 EPROM (PROGRAMMED)
1 - 7805 regulator

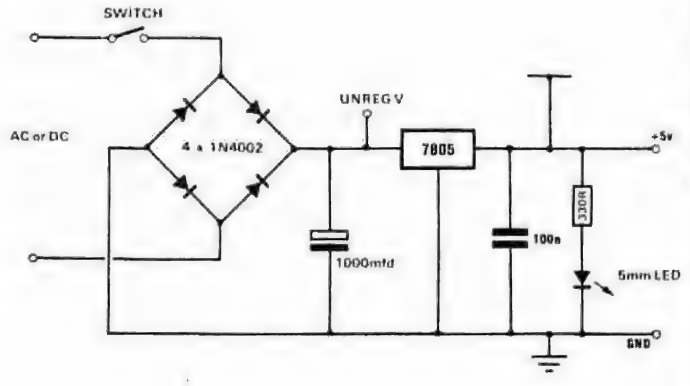
1 - 20 pin IC socket
1 - 24 pin IC socket
1 - 40 pin IC socket
1 - 8 way DIP switch
1 - DPDT slide switch
3 - PC mount push switches

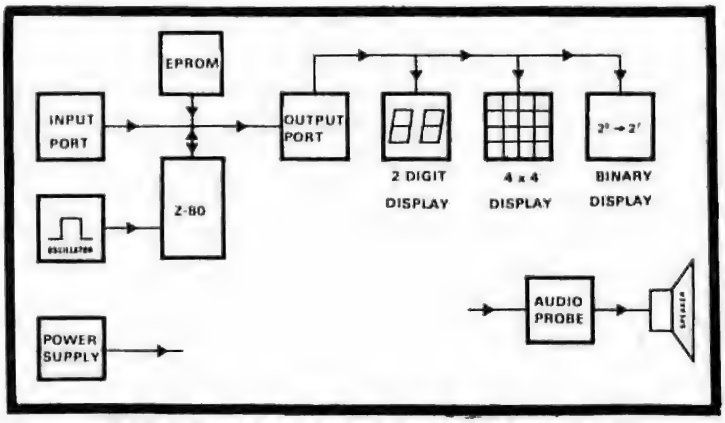
1 - 3.5mm mono socket
1 - mini speaker

1 - 6BA nut and bolt
4 - rubber feet
13 - matrix pins
1 - hollow pin
20cm hook-up flex
1 metre tinned copper wire
1 - female matrix pin connector
2cm heatshrink tubing.

MICROCOMP PC BOARD







BLOCK DIAGRAM OF MICROCOMP

CONSTRUCTION

Lay all the components on a sheet of paper and identify them. Make sure all parts are present.

Start assembly by fitting the jumper links. There are 41 of them and each must be inserted carefully to produce a neat result. For each, cut a piece of tinned copper wire longer than required and bend it to form a staple, with the long lower section kept as straight as possible. The two ends must fit down the holes cleanly and the wire must be able to be pushed right up to the board. This means the bends must be sharp.

If the two ends do not protrude through the board, do not attempt to solder the link as this will produce a dry joint which will be very hard to locate when troubleshooting. We have had two cases of this and it took hours to locate the fault.

Solder the ends of each jumper and cut the ends off with a pair of side cutters so that a little of the wire emerges from the solder. Do not cut through the solder as this will fracture the joint and possibly cause a fault.

Next fit the IC sockets. Make sure each pin fits down a hole before starting to solder. If a pin bends under a socket it will be very hard to rectify after the socket has been soldered. So check before-hand.

Solder one pin at each end to keep the socket in place while you attend to each pin.

Solder each pin very quickly and use fresh solder for each connection. The solder mask prevents the solder running along the leads or touching any of the lines which pass between the pins. It helps give a professional result and makes your soldering 100% neater. But don't use too much solder or blobs will result.

On the other hand don't use too little or the leads will not be fully surrounded by solder.

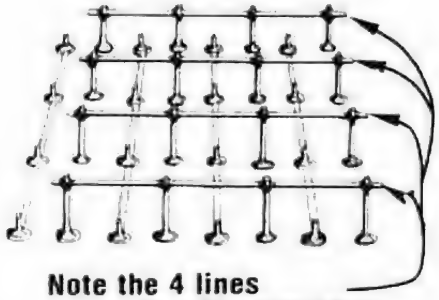
All the components are marked on the PC board and it is possible to build the project without any other help. But as a guide we will go through the assembly and explain everything as we go.

Basically you start with the smallest components which are closest to the board and progress to the highest or largest components.

We have fitted the lowest items and now the smallest. There are 13 connector pins on the board and one hollow pin for the TONE OUTPUT. The pins accept flying leads and female connectors as used in the plug-in modules.

Fit the 16 LEDs in the 4x4 display and 8 LEDs in the single row so that the flat on the side of each LED is on the right. Refer to the markings on the PC board. The cathode leads of the LEDs in the 4x4 are left long and a piece of tinned copper wire soldered across them, about half a cm from the board.

There are 4 individual pieces of tinned copper wire which join the cathode leads of the LEDs to the circuit. Refer to the drawing to see how this is done.



Note the 4 lines connecting the cathodes:

Next add the resistors and signal diode in the clock circuit. All these components touch the board and the leads are trimmed neatly after each is soldered. Next fit the 4 power diodes and eight BC 547 transistors. Almost any NPN small signal transistor will be suitable and BC 547 is only used as a guide. There is one BC 557 transistor used as the input decoder and this is indicated on the board with a white transistor symbol. All transistors should be pushed onto the board leaving a space between body and board equal to about the thickness of a resistor.

Mount the 100k mini trim pot and solder its leads. Push the leads of a LED through the screwdriver slot in the pot and bend them over so that the body becomes a handle. By turning the LED you will notice the trim pot rotates too.

Now comes the need for a careful bit of soldering. The two leads of the LED must be soldered to the rotating part of the pot so that the solder does not run over the edge and touch any other parts. If this happens the pot will be ruined as it will no longer rotate.

Fit the two 1n greencaps into the clock circuit.

Mount the ON-OFF switch and input jack so that they touch the PC board and solder the leads carefully.

The 7805 regulator is mounted under the PC board with a nut and bolt so that it touches the copper laminate. This will act as a heat sink and prevent the regulator getting too hot.

The leads from the regulator fit into the holes on the underside of the board and are snipped off the top side so that they don't protrude.

The two electrolytics must be mounted around the correct way. Observe the negative marking on the component and the positive marking on the board. The 1mfd reset electro is bent over and lays flat on the board to prevent it getting in the way of the reset button. Allow enough lead for this to be done.

A 'POWER-ON' LED is fitted near the regulator to indicate 5v.

Three push buttons are the next components to be fitted. The positioning of these is determined by a flat on the side of the switch aligning with the marking on the PC board. You can use any colour for the switches as they are not colour coded.

The mini speaker can be mounted either way around as it is not polarity sensitive. A 10cm wander lead is required for the probe and it must be long enough to reach over the entire board. A short piece of stiff wire can be soldered to the end of the lead to act as a probe tip or alternatively the wires can be soldered to make them stiff.

One jumper lead is required on the board to select either the upper half of the 2732 or lower half. A female socket is attached to the lead and kept in position with a short piece of heat-shrink tubing.

The last component to be fitted is the 8 way DIP switch. The numbers and/or letters on this switch must be removed before it is fitted to the board as they are not used in this project and may cause confusion.

Use a knife or blade and scrape the numbers until they disappear. Next you must determine which way around the switch is to be inserted as some switches are CLOSED when the lever is UP while others are closed when the lever is DOWN.

We require the switch to be closed or ON when the lever is DOWN so that each of the levers correspond to a number on the PC board. This is not essential and the switch will work satisfactorily around the other way, but to make things simple keep to our suggestions.

Check the operation of the switch with a multimeter before inserting it onto the board and solder it in position when it is correct.

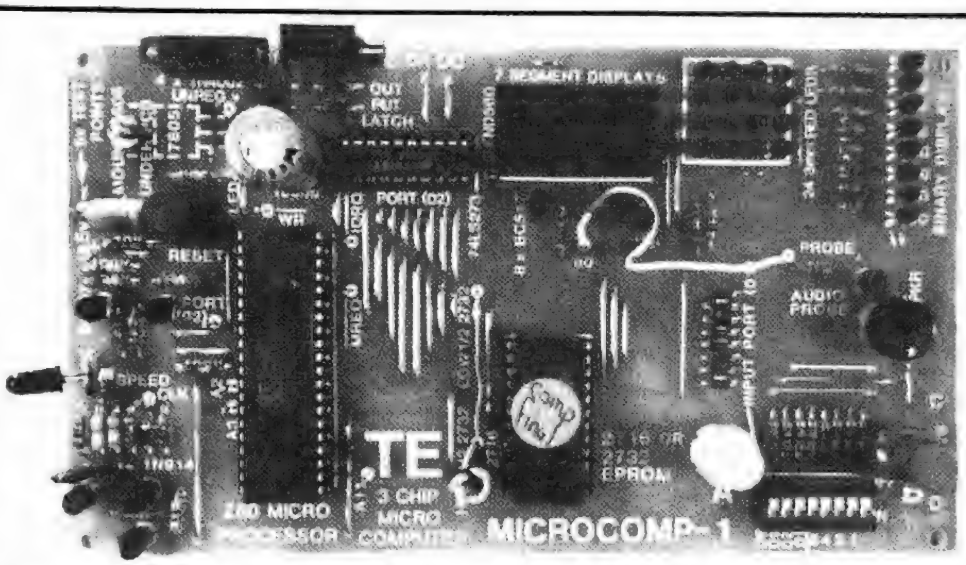
Fit 4 rubber feet to the underside of the board, insert the chips and you are ready for testing.

TESTING

Insert the power plug into the 3.5mm socket and switch the Microcomp ON. The power LED should come on. Make sure all the input switches are OFF. Push button B. The number 99 should appear on the displays. Press button A and the numbers will increment. Push button B and they will decrement. This is a fairly good indication that everything is working perfectly and you can go on to learning about programming.

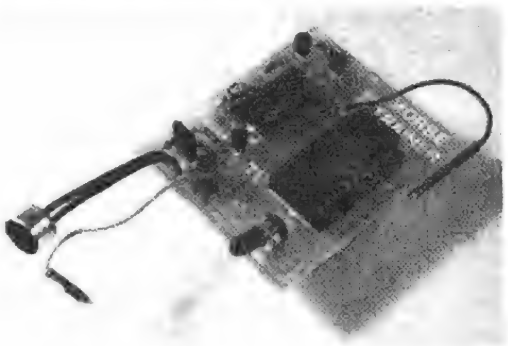
If you do not get 99 on the displays you may have a fault in the system. This will require you going through a trouble-shooting procedure as covered on P. 66.

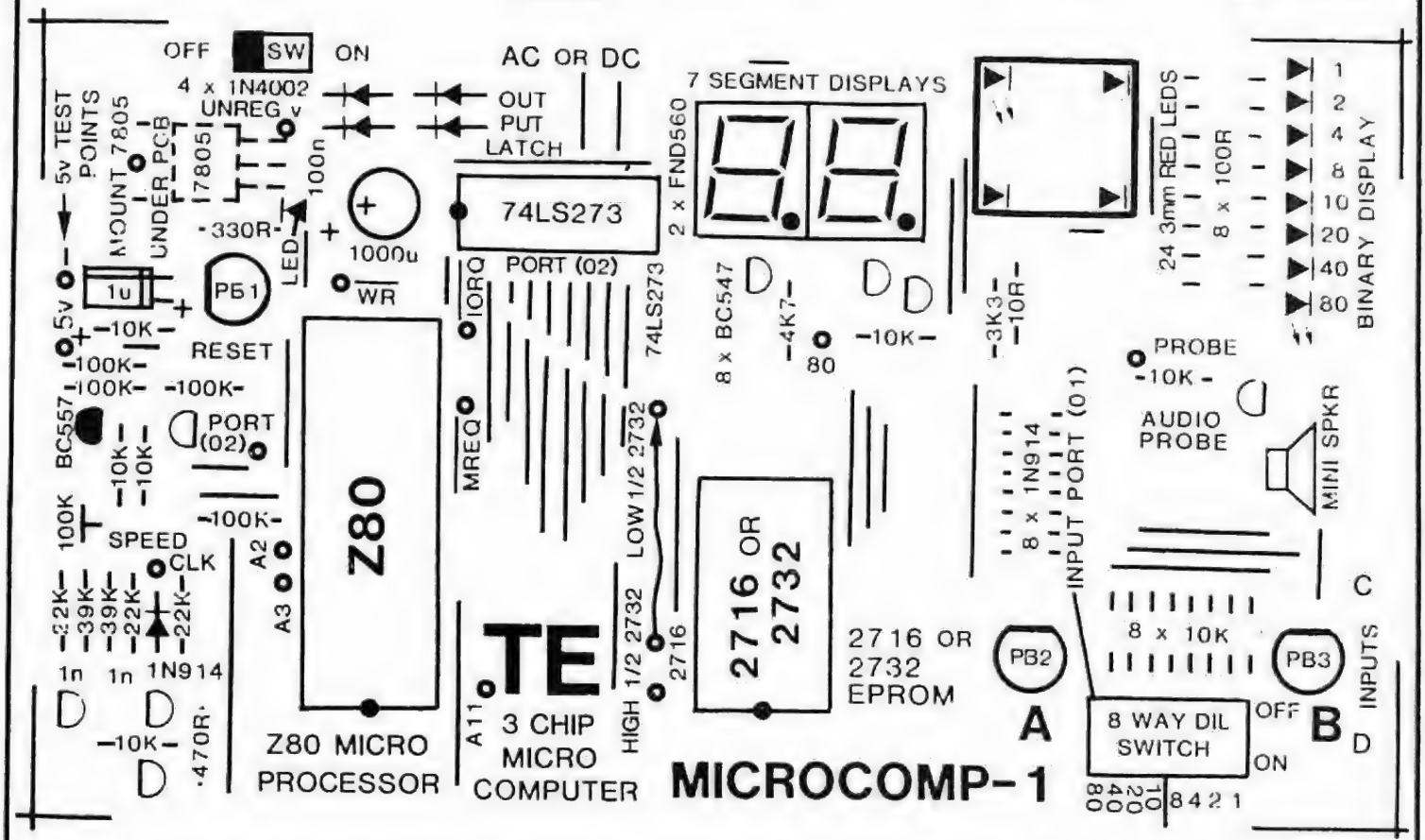
Consider yourself lucky that the computer doesn't work. You will gain a lot by trouble-shooting it yourself and gain experience in finding the fault.



Note the LED used as a knob for the SPEED control. SGS transistors don't work very well in the clock circuit. They freeze at high speed. To prevent this, use 47k base resistors.

The MORSE TRAINER is our first add-on and will be presented as soon as the programs in the lower half of the 2732 have been covered.





The overlay for the Microcomp shows all the component locations and link positions. The large donuts indicate the positions for the matrix pins. A wander lead selects HIGH/LOW 2732, while another is taken from the AUDIO PROBE input pin.

IF IT DOESN'T WORK

As we have said, digital projects are extremely reliable and have an enormous success rate. The chances of this project working as soon as the power is applied, is very high.

However, if it doesn't snap into life, here are some helpful suggestions.

Firstly check the power LED. It should come on as soon as the power is applied. If it doesn't, the fault will lie somewhere in the power supply.

Feel the 7805. It should get warm after operating for a few minutes. If it is very hot, you will have a short in the circuit. Turn off the computer and look for a bridge between two tracks. This may be anywhere at all on the board and this is how to go about it:

Measure between the positive and negative rails with an ohm-meter set to LOW OHMS. It should measure about 30 ohms in one direction and 50 ohms in the other. The values you will get are mainly due to the presence of protection diodes inside the chips and the resistors on the board. The actual value of resistance does not matter. Values such as this do not indicate a short circuit. But if it 10 ohms or less, a short-circuit is present.

Remove one chip at a time. If the low value is still present after all the chips have been removed, you will have to look for a fault on the board itself.

Start by removing the 7805 and then one end of each of the 41 jumper links. Measure the resistance value at each stage. If the short is still present, lift one end of each resistor and capacitor. If it is still there, it will possibly be a short between 2 tracks. You will need a magnifying glass and a sharp knife. Cut between the tracks at every location where you have made a connection to make sure no wiskers of solder are shorting between one land and another.

After this, the short should be removed. Refit the 7805 and switch ON. The LED should light. Refit all the components and jumpers. Use desolder wick to remove the solder from each of the holes so that the leads can be inserted.

If the power LED comes ON but none of the displays, set an input value on the switches of say '40' and reset the computer. This will produce '99' on the displays. If they remain blank, you will have to look into the operation of the system.

This is where the built-in **AUDIO PROBE** comes in. The probe lead will enable you to hear the signals on each of the active pins of the chips. We have specially designed the computer to operate at a speed which can be heard by the human ear. The probe will let you hear the frequency of the clock, the output of the address and data bus and also the activation of the latch.

Firstly turn the clock speed down and probe the 'clock-input' at pin 6 of the Z-80. You should hear a fairly high pitched whistle. As you increase the clock speed, this whistle will increase until it gets too high to hear. Next probe one of the data lines and you will hear a tone which is exactly one-eighth the frequency of the clock. If nothing is heard, it means the Z-80 is not operating or not accepting the input clock waveform. Make sure the reset pin of the Z-80, pin 26, is HIGH, otherwise the Z-80 will be sitting in a reset state.

If nothing is heard on the address or data buses, the fault will lie between the Z-80 and EPROM. They must be talking to one another for the system to start up. Even a blank EPROM (filled with junk of FF's) will produce a tone on the buses.

Test pin 18 of the EPROM to make sure it is being accessed. You should hear a tone on this pin which means the Z-80 is accessing the EPROM and trying to get it to place data on the data bus.

Some of the faults which can occur between the Z-80 and EPROM include non-soldered connections, IC socket pins which do not pass through the PC board and thus do not connect to the circuit, power not reaching the chips (due to a broken track), or a fault in one of the address or data lines near a solder connection.

This generally occurs when you are soldering and may be due to the iron being too hot, taking too long to produce the joint or moving the component while the solder is setting. The result is a hairline crack where the track meets the land and this is very hard to spot. Use a multi-meter set to low ohms to measure the continuity of each of the lines.

If everything seems to be correct, try replacing the Z-80. It does not matter if you use a Z-80 or Z-80A, they will both work equally well.

There is only one remaining possibility. The Z-80 requires a perfect square wave for it to function and we have gone to a lot of trouble to produce a near perfect waveform.

If the rise and fall time is not extremely short, the Z-80 will not accept it. This problem will be almost impossible to determine, even with a 30MHz CRO. If you have come to this conclusion, you should send your project in for a check-up.

Once you have values appearing on the displays, you can check for the correct operation of the programs by accessing our OUTPUT LATCH TEST ROUTINE. Turn on switches 01, 08 and 20. This will give a value of 290. Push reset and the micro will jump to address 0290. Three LEDs should illuminate: 01, 08 and 20. Now turn all switches OFF. All LEDs should extinguish. If any remain ON, the fault could lie in the input port. Check the soldering for shorts and all lines for continuity.

If a fault is present in one of the lines other than 01, 08 or 20, the micro will not address 0290 and the wrong program will appear.

If this is the case you will have to experiment with various settings and try to determine where the micro is jumping to.

If a wrong program is picked up, you cannot be sure it has accessed the beginning of the program and thus you cannot immediately determine which line is at fault.

Turn all switches OFF and press reset. The computer should not address any programs as the jump routine will be loading HL with 00 00 and jumping to the start again. Thus it will run around a loop, back to address zero.

If the 7-segment displays illuminate but not the 4x4 matrix, or the row of 8 LEDs, the fault will lie in the jumper lines which must be connected to the cathode leads of each of the 16 LEDs. See the construction notes for this as it will be the first time you have come across this method wiring the underside of a board.

The decoding transistors for ports 1 and 2 only come into operation when they receive the correct instruction via the program.

When the micro is executing the start-up program, it will be looking at the input port twice per loop and you will able to hear this in the mini speaker.

The output port will not be accessed during this time and probing the Latch Enable line will give no tone.

You must put a value on the input port switches to get the computer out of the start-up routine if you want to probe the output decoding transistor.

This is done at pin 11 of the 74LS273. If no tone is heard, trace the circuit back to the BC 547 (near the Z-80) and probe the base and emitter leads. When the transistor is being turned ON, a tone will be heard in the collector circuit.

If all these suggestions fail, start at the beginning again and solder each connection. Use desolder braid to collect any excess solder and inspect every joint under a bright light.

Make a continuity check of each copper track and make sure each land is not shorting to the one next to it.

Check all the LEDs for correct insertion and all chips for placement around the correct way.

Check the regulator, the 4 power diodes, the clock circuit, the placement of the 9 transistors, the positioning of the 3 push buttons and the value of all the resistors.

Ask a friend to go over the project and carry out the troubleshooting hints.

If all this fails, there is a repair service from TE and for $15.00 plus $4.50 postage, you can get your unit repaired. Send it in a jiffy bag with $19.50 and we will do our best. Up to now every computer sent to us has been repairable.

So, don't despair. Send it in and we'll check it out.

WHAT EACH CHIP DOES

There are 5 major building blocks in the MICRO-COMP. They are:

THE CLOCK - made up of 3 transistors
THE CENTRAL PROCESSOR UNIT - A Z-80.
THE MEMORY - A 2732 EPROM.
THE INPUT PORT - 8-way DIP SWITCH
THE OUTPUT LATCH a 74LS273

There are also a number of other active devices (transistors) which perform inverting and driving functions and also a single transistor connected to a mini speaker to provide an audio probe to listen to the computer in operation.

We have intentionally kept the chip-count down to make the project attractive and in this chapter we will discuss each chip and how it fits into the circuit.

THE CLOCK

Even though this is not a chip, we could have used one. The requirement of a clock is to produce a very fast rise-time waveform at a frequency to suit the project.

The clock in a computer controls the speed at which data flows through the whole system. The Z-80 will operate at a frequency as low as 7kHz and below this its registers will fail to hold information. This is because they are dynamic and have to be 'topped up' many times per second.

At the higher end of the range, the Z-80 will operate at 2.5MHz and a Z-80A at 4MHz.

In our project, we want the Z-80 to operate as slow as possible so that we can 'see' the program run and hopefully listen to the bus lines change tone as the program runs through its steps.

The reason for the clock circuit containing a diode and wave-shaping transistor is to generate a perfect square wave. The Z-80 is very critical as to the shape of the wave it will accept and the rise and fall edges must be extremely fast - especially at this low frequency.

In addition, we have included a speed control in the clock circuit so that the frequency can be adjusted from 7.5kHz to 35kHz. This is nearly a 5:1 ratio and allows each of the programs to be run at high and low speed.

Even at these speeds the Microcomp must be one of the slowest computers on the market as most operate at a clock frequency of 1MHz to 2MHz. But don't worry, even at 8kHz, you will see operations performed faster than you can think.

THE Z-80

This chip is the heart of the computer. It is called the CENTRAL PROCESSING UNIT or CPU for short. This is a truly an amazing chip and we could fill many pages on its workings.

You will pick up a lot more on how the Z-80 works as we progress with the notes and the main fact is it controls all the other chips in the system. It takes information from the 2732 and delivers the result of calculations and operations to the output latch. The speed with which it performs these tasks is controlled by the frequency of the clock.

The Z-80 is capable of controlling over 100 chips and you can see our 'comp is only a very small design.

The Z-80 is like a story-teller. It reads the 2732 like a book and delivers its interpretation to a child (the output latch). The input port is like a child telling the story-teller where to start in the book. The clock circuit is like a watch - telling the story-teller how fast to read.

THE EPROM

Chip number two is the program storage chip. It has been programmed by TE so that a number of programs and effects can be produced on the displays. These chips are bought in a blank condition and programmed by means of an EPROM PROGRAMMER so that they contain the necessary set of HIGHs and LOWs to make the Z-80 perform the required operations.

The EPROM supplied in the kit is ready to operate the computer but you can program your own or get a friend to program one for you and it will work just as successfully. The full listing to do this is supplied in the notes.

This is the main advantage behind the type of programming we are covering. It means you will be able to write programs for your own micro-computer controllers, produce the EPROM and get it running without the need for any outside help.

It is the most efficient type of program available, in terms of memory required. It consumes the least amount of memory and is used in all types of industrial applications and video games.

THE INPUT PORT

This is an interface between the computer and the real world. We have already mentioned the need for this connecting link.

The input port takes in information from a set of switches and loads it into the accumulator in the Z-80. The Z-80 operates on this according to the instructions in the program.

As well as the 8 switches, there are also 2 push buttons which are in parallel with the two highest value switches. Provision for two more switches (external to the board) is also provided on the PC.

The input port is software controlled and thus any of the switches can be programmed to perform any operation you wish. They can start a program, stop it, call up a number, increment a count value, decrement it, sound an alarm, dial a phone number and lots more.

A switch places a HIGH on the data bus, when it is closed, and only when instructed to do so via the program. The instruction for this is: IN A,(01) and the input decoder transistor is activated to allow this loading to take place. At all other times the switches put no load on the bus and allow the lines to rise and fall so that the other instructions in the program can be performed.

THE OUTPUT LATCH

The output latch is the third and final chip in the project. This is the chip which drives the set of output LEDs and displays. We have created three different types of display and each will produce its own special effect according to the program being run. The main purpose of this latch is to hold the information coming from the Z-80 for long periods of time so that we can view it on the displays. This allows the Z-80 to go away and carry out other operations.

A set of transistors turn on one or other of the 7-segment displays via the 8th line so that a two digit number can be displayed.

INPUT/OUTPUT

The Microcomp is capable of accepting information from the outside world as well as delivering to the outside. This capability is called INPUT/OUTPUT.

In a simple system such as ours, for each address line it is possible to connect 8 devices to the data bus and access them individually via the program. These devices must also be gated into operation via the IORQ line.

Devices can have either input or output capability and since the Z-80 has 16 lines, this gives us a lot of devices! This is more than we require and to keep it simple we will consider only one set of 8 on address line A0 and one set on address line A1.

THE INPUT PORT

Input information is obtained from a set of 8 DIP switches and these are connected to the data bus. Eight switches like this gives us the capability of up to 256 combinations.

When address line A0 goes HIGH and IORQ goes LOW the value on these switches is passed to the Z-80 as an input value.

These switches are software programmable and can be instructed to perform many tasks, depending on the instructions in the program. The micro only looks at the switches during the instruction IN A,(01) and during the remainder of the time the switches are allowed to float up and down and don't interfere with the data bus.

THE OUTPUT PORT

The OUTPUT PORT is a latch chip. It must be a latch to hold the output value long enough for us to see the data on the displays. The latch will retain this data until updated.

There are two gating transistors in this project. One controls the input port and the other controls the output port.

Each transistor produces a LOW output when the I/O Request is LOW and the prescribed address line is HIGH.

The I/O Request line does not determine the IN or OUT nature of the signal, it just goes low when the Z-80 requests one of its ports. The circuitry and instruction in the program determines the IN or OUT condition.

THE DISPLAYS

The Microcomp has three different types of displays:

★ **Two 7-segment displays**
★ **A 4x4 matrix of LEDs**
★ **A row of 8 LEDs.**

Each display provides a different effect for any given set of values and you will be able to make a comparison between them as the programs run.

Here are a few facts and hints on producing effects on the displays.

At first you may be surprised to see two 7-segment displays operating from one latch chip. Normally this is not possible as all the lines from one latch are required to drive the LEDs in the display.

But by using only 7 lines to drive the segments, we have one line left over to switch between the two displays. This eighth line is normally used to drive the decimal point but this is the sacrifice we have had to make.

In our arrangement only one display will illuminate at a time and to make them both appear to be illuminated at the same time we must switch rapidly between them. This will create a two-digit number and allow us to produce a readout for a 00 to 99 COUNTER. It will also give us a number of other effects as you will see in the programs.

The 4x4 also connects to the latch and because the LEDs are connected in a different way to the 7-segment displays, a completely different effect will be created. A program for the 4x4 will not be recognisable on the 7-segment displays and vise versa.

The 4x4 matrix can be thought of as a miniature display board. It is connected to the latch via 4 horizontal lines and 4 vertical lines. The anodes of the LEDs are connected to the 4 lower bits of the latch such that the first column goes to bit 0. Column 2 goes to bit 1, column 3 to bit 2 and column 4 to bit 3.

The anodes of all the LEDs are connected to the 4 higher bits of the latch such that the lowest row connects to bit 4. The second row connects to bit 5, the third row connects to bit 6 and the top row to bit 7.

This means bit 0 sources 4 LEDs and so does bit 1, 2 and 3. Bit 4 sinks 4 LEDs and so does bit 5, 6 and 7.

To turn on a LED, the source bit must be HIGH and the sink bit must be LOW. This arrangement will allow any individual LED to be illuminated and even certain combinations of LEDs. But it does not permit absolutely any combination to be illuminated due to our wiring.

We can overcome this by a trick in programming called multi-plexing. This will be covered later and can be seen in the dice project.

To see exactly how the LEDs are accessed, address the program at 0290. By switching off the input switches you will turn the matrix off. Load input values into the switches and you will see the rows and columns of LEDs illuminate.

The third display is a row of 8 LEDs. This display can be referred to at any time for both the binary value and hex value being outputted from the latch. The binary value is simply obtained by looking along the row of LEDs and noting the on-off pattern. By adding their value in binary you obtain the decimal value of the latch.

But decimal values are of no real use to us in this project as we are concentrating on hexadecimal notation.

To find the hex value of the output latch, add the hex values alongside each LED. This is easy to do after a little practice.

Using the three displays together you will see the hex value required to produce letters and numbers on the

7-segment displays and also see what the micro in inputting and outputting in binary form to create these numbers and letters.

In all, it gives a graphic picture of what is going on.

THE AUDIO PROBE

The audio probe consists of a single transistor and a mini speaker. Its prime function is to enable you to listen to the 'computer in operation'.

This is possible when the clock speed is turned down and the probe touched on each of the pins of the chips.

It is interesting to hear the HIGHs being sent along the lines, especially the address bus where each line is running at half the frequency of the previous. The Z-80 is acting like a 16-stage divider and you can hear this on the probe.

The probe is also used for determining the operation or non-operation of the Z-80. This is one of the tests you will be required to do when setting up the project as the Z-80 requires a near-perfect square wave for it to operate.

The easiest way to see if it is accepting the clock pulses is to listen to the address or data lines.

The only way to know if the Z-80 is accepting the clock is to use the probe on pin 6 of the Z-80 and then on one of the address lines.

The audio probe is also used during the course of the experiments. By comparing the program with the tones on the buses and the Latch Enable pin, you can determine how often the chip is being accessed.

The audio probe also connects to pin '80' on the PC board which is bit 7 of the output latch. The Tone program at 0010 outputs a HIGH to this line and then a LOW to produce a click in the speaker. This is the basis to producing tones and by varying the speed control, the pitch can be altered.

WHAT IS THE 2732?

The 2732 is a memory chip containing 32,768 individual cells which can be programmed to contain a small charge.

Each cell is a single P-channel MOS transistor capable of detecting the presence of a charge.

This charge is held on a conducting layer above the transistor, on a thin film of insulating material. When the charge is present the transistor outputs a HIGH. When the charge is not present, the transistor outputs a LOW.

We can access each of these 32,768 cells and supply them with a small charge during programming. The charge remains in place for many years because it has no where to jump to as each area is surrounded by insulation.

Exposure to ultra violet light will give the charges sufficient energy to jump off, leaving the plate in a neutral state.

When you look through the quartz window you can see the array of cells. It seems incredible that over 32,000 cells can be seen, but that's the reality of electronics.

We access these cells 8 at a time and this is equal to one BYTE. This is the basic unit which is fed into a processor and is the basis of all Machine Code programs.

One byte can have up to 256 possibilities due to the fact that each of the 8 cells can be either ON or OFF.

To output these 8 bits of data from the chip we need 8 lines and these form the DATA BUS.

We need another set of lines into the chip so that we can locate these 8 cells. For a 2732 we need 12 lines and these are called the address bus.

There is one interesting feature about the address and data lines. Even though they are identified as A0, A1, A2, ...D0, D1, D2 etc. they can be connected to the microprocessor in any order. This is because the cells are uncommitted and provided you read in the same order as it was programmed, the correct data will be outputted.

The only reason for keeping to an accepted pin-out arrangement is so the EPROM will work in other designs and on common programming equipment.

THE GATING TRANSISTORS

Input and output ports must only come into operation when requested. At all other times they must not put a load on the data bus as it is required for other communications.

However when an instruction such as IN port 1 is sent to the Z-80, there are two lines which will be held in a stable condition and can be used to activate the port latch. These are I/O Request and address line A0. These can be gated together and the resulting pulse used to activate the port.

This is called simple decoding and since the Z-80 has a number of address lines it is possible to connect lots of input/output devices.

We have used only the first two lines, A0 and A1 and they provide a simple way to achieve an end result.

With this arrangement, the first device will be activated with the instruction: IN A,(01) and the second by IN A,(02). Further devices would be activated via IN A,(04) IN A,(08) and IN (10),A. By adding port values together, more than one port can be activated at the same time, should this be necessary.

USING THE DIP SWITCHES

The 8 dip switches are connected to the input port and are capable of providing up to 256 different combinations.

Eight lines like this is equal to one byte and depending on the program being run, this value can be used in many ways. Examples can be seen in the programs contained in the EPROM that comes in the kit.

We will now explain the meaning of the values on the PC board, alongside each of the switches.

You will see numbers: 80, 40, 20, 10, 8, 4, 2, 1. These are hex values and are an easy way for us to give values to a set of binary switches. The other option is to write: 1, 1, 1, 1, 1, 1, 1, 1.

Hex is a successful solution to writing values from 1 to 256 in a form which is easy to read and only requires 2 digits. To input a value such as 234 refer to the Hex Conversion table on P 16 of issue 11. It is equivalent to EA. Once you are in Hex notation, you stay in Hex. This makes it awkward when you see values such as 10, 20 45, 80, 100 but you must remember these are also Hex values and a number such as 10 (one-oh) is really 16 in decimal notation.

To place EA on the switches, you need to know about Hex addition. For instance E is made up of: 8, 4, 2, and 1. This is how it is done on the input switches: The switches are separated into two banks of four. The low value switches are labelled 8, 4, 2, 1. The high value switches are 80, 40, 20, 10.

The value EA is placed so that E will be loaded into the high section and A into the low section. To enter E turn

on switches 80, 40 and 20. This gives E0. To produce the value A, turn on switches 8 and 2. The input switches now hold EA.

After you have used them a few times you will become familiar with their operation.

One of the main uses is to generate a JUMP VALUE to get to the programs in EPROM. The computer interprets the value on the switches as a START ADDRESS by multiplying the value by 10 (one-oh) and jumping to the address of the value created.

The multiplication value of 10 is a hex value and is equal to 16 in decimal.

For example if we load the switches with the value '1', the start-up program will convert it to 10 and produce the address 0010. This is the address of the first program in memory - a TONE routine. To address the RUNNING NAMES routine, load the switches with 8. This will make the Z-80 jump to 0080, when the reset button is pressed.

In a similar way, the start of each of the programs can be accessed via the switches. For instance, the Final Message at 07A0 is addressed by loading 7A.

Although we can only address every 16th location, the programs have been written to start at an even Hex value and end before an addressable location. Some programs occupy 80 or more bytes while other take less than 8. This means some locations will be unused but this is the limitation of the system.

Experiment by loading the start address of various programs and run them to see how they operate.

THE PROGRAMS

We now come to the programs themselves.

The list shows all the programs in the lower half of the 2732. The number in the first column is the START ADDRESS which is loaded into the DIP switches. Once the program has been accessed, you can use the push buttons and any of the DIP switches to operate the program.

Whether you have burnt your own EPROM from the listing or bought a kit, you will want to know how the programs are put together and how they run. That's the whole purpose of this project.

Study each program carefully, running it at different speeds and answer any questions associated with the listing.

LIST OF PROGRAMS:

0000	JUMP ROUTINE
0010	TONE
0020	QUICK DRAW
0080	RUNNING NAMES
00D0 - 00F4	RUNNING LETTER ROUTINE (can be called)
100 - 1FF	LIST OF NAMES
200 - 28F	LOOKING AT DATA
290 - 29F	FROM INPUT TO 8 LEDs
2A0 - 2BF	INCREMENT VIA BUTTON A
2C0 - 2CC	AUTO INCREMENT (fast)
2D0 - 2DD	AUTO INCREMENT (variable)
2E0 - 2EC	AUTO DECREMENT
2F0 - 2FF	AUTO DECREMENT (variable)
300 - 36F	4x4 EFFECTS
370 -	0 - 9 COUNTER
390 -	0 - F COUNTER
3A0 -	A - Z, 0 - F COUNTER
3F0 - 3FF	VERY LONG DELAY
400 - 469	00 - 99 COUNTER
470 - 51F	DICE
520 - 52F	EPROM IN BINARY
530 - 623	POKER
630 - 6BF	BINARY CLOCK
6C0 - 6CB	ONE MINUTE TIMER
6D0 - 6DB	3 MINUTE TIMER
6E0 - 6EB	1 HOUR TIMER
6F0 - 738	ADJUSTABLE TIMER
740 - 760	1 MINUTE DELAY
765 - 79D	Table for adjustable Timer
7A0 - 7FF	FINAL MESSAGE

These programs occupy the lower ½ of a 2732 EPROM.

at 0000:
THE JUMP ROUTINE

This routine will be used every time you want to access one of the programs.

Set the address value on the input switches and press reset. The micro will then jump to the program you have selected.

Each program is a loop and the Microcomp will run around this loop.

The input switches can now be used for other functions according to the demands of the program. Don't push reset as this will cause the micro to jump out of the program. Only buttons A and B are used during the course of the programs. These are equivalent to switches 8 and 7.

THE JUMP PROGRAM

```
1. LD B,00        0000  06 00
2. IN A,(01)      002   DB 01
3. LD HL 00 00    004   21 00 00
4. LD L,A         007   6F
5. ADD HL,HL      008   29
6. ADD HL,HL      009   29
7. ADD HL,HL      00A   29
8. ADD HL,HL      00B   29
9. JP (HL)        00C   E9
```

This routine looks at the input port (01) and jumps to the address set on the input switches.

The program multiplies the value set on the switches by 10 (one-oh) and jumps to this value.

If no switches are set, the program constantly loops back to 0000, looking for an input from the switches.

If '1' is loaded on the switches, the program jumps to 0010. If '2' is set, to program jumps to 0020 etc. If switches 20, 8 and 1 are set, the program jumps to 0290.

In this way we can access from 0010 to 07F0 in blocks of 10 hex bytes. This is equal to every 16 bytes and gives us a very good coverage of the EPROM.

The way in which the program works is this:

Line 1 loads the B register with 00 ready for a **DJNZ** statement as required in some of the programs. It has nothing to do with this program.
Line 2. The program looks at the input port and loads the value it finds on the switches into the accumulator.
Line 3. The HL register pair is zeroed.
Line 4. The accumulator is loaded

into the L register, which is the LOW register of the pair.
Lines 5, 6, 7 and 8 add the contents of the HL register pair to itself four times. Each **ADD** doubles the result, making a total increment of 16 times. A multiple of 16 is equal to **10** in hex.

Line 9. The micro jumps to the address given by the value of the HL register pair.

QUESTIONS:

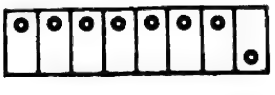
1. Set the switches to address values which are not the start addresses of a program. Why do some of them work?
2. Why does button B address the start of the 00 - 99 counter?
3. Could the DIP switches be replaced with push buttons?
4. Explain what we mean by the input switches are software programmed:
5. Name a few devices which can be connected to the input port:

ANSWERS

1. Sometimes you can start part-way through a program and it will run. This is because the micro jumps into a location it understands and it follows the program to the end. It then jumps to the start of the program and produces a full display on the screen.
2. Button B has the same value as '40' on the switches and this corresponds to address 400 in the EPROM.
3. Yes, but remember up to seven buttons would have to be pressed at the same time to achieve the result of the DIP switches.
4. The input switches can be programmed to do anything, as requested by the program.
5. Any device which has a set of contacts such as a relay, morse key, micro-switch, pressure mat or even transistors acting as switches can be used.

THE TONE ROUTINE

The TONE routine is located at 0010 and this is addressed by switching the lowest value switch ON thus:



The principle behind creating a tone is to toggle an output bit. The speed with which the bit is toggled, produces the frequency of the tone. To produce a 1kHz tone requires a minimum clock frequency of about 50kHz. This is because the clock frequency is divided by eight to run the data bus and further clock cycles are required for the load and output instructions. Since the maximum frequency of the Microcomp is about 35kHz, the highest tone which can be produced is 700Hz.

This is not sufficient for a musical scale or a tone generator and only a sample tone has been included in the EPROM.

By inserting the lead of the AUDIO PROBE into terminal '80' on the board, below the 7-segment displays, the tone will be reproduced in the mini speaker.

You can compare this tone with the Latch Enable pin and the data bus and see if the tones are different.

The TONE routine is a loop, starting at 0010 and ending at 001F. The first instruction **AF** is a single byte instruction which clears (zeros) the accumulator so that this value can be outputted to port 2. The accumulator is then loaded with 81 which

TONE ROUTINE:

XOR A	**0010**	**AF**
OUT (02),A	**0011**	**D3 02**
LD A, 81	**0013**	**3E 81**
OUT (02),A	**0015**	**D3 02**
XOR A	**0017**	**AF**
OUT (02),A	**0018**	**D3 02**
LD A, 81	**001A**	**3E 81**
OUT (02),A	**001C**	**D3 02**
JR 0010	**001E**	**18 F0**

produces a HIGH to the AUDIO PROBE input pin and also turns on segment 'a' of the first display. This is the complete TONE routine. The sequence has been repeated again to use up the available memory before jumping back to 0010 via a JUMP RELATIVE instruction.

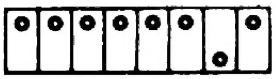
The program will loop continually until the reset button is pressed. The input switch must be OFF to prevent the program being accessed again.

QUICK DRAW PROGRAM

LD C,02	**0020**	**0E 02**	This is the COUNT register for 2 rotations of the display.
LD D,08	**0022**	**16 08**	D is the COUNT register for the 8 LEDs
LD HL,00F5	**0024**	**21 F5 00**	Load HL with the start of the byte table.
LD A,(HL)	**0027**	**7E**	Load the accumulator with the value POINTED TO by HL.
OUT (02),A	**0028**	**D3 02**	Output the accumulator to port 2.
DJNZ 002A	**002A**	**10 FE**	Register B contains 00 (via jump program) **DJNZ** is a DELAY.
INC HL	**002C**	**23**	Increment HL to look at the second byte in the table.
DEC D	**002D**	**15**	Increment the BYTE-TABLE COUNTER.
JR NZ 0027	**002E**	**20 F7**	If end of table not reached, jump to line 4. Otherwise next line.
DEC C	**0030**	**0D**	Decrement C and illuminate 8 LEDs again.
JR NZ 0022	**0031**	**20 EF**	If C is zero, advance to next line.
LD A,00	**0033**	**3E 00**	The accumulator is zeroed to blank the display.
OUT (02),A	**0035**	**D3 02**	The Accumulator is outputted to port 2
LD DE, 0602	**0037**	**11 02 06**	The DE register pair is available for a long DELAY.
DEC DE	**003A**	**1B**	Decrement DE
LD A,D	**003B**	**7A**	Load D into A
OR E	**003C**	**B3**	OR E with the accumulator
JR NZ 003A	**003D**	**20 FB**	Jump if both D and E are not zero.
IN A,(01)	**003F**	**DB 01**	Input the two switches.
BIT 6,A	**0041**	**CB 77**	Test BIT 6 to see if switch B is pressed.
JP NZ 0020	**0043**	**C2 20 00**	Jump to start of program if button B is pressed.
BIT 7,A	**0046**	**CB 7F**	Test BIT 7 to see if button A is pressed.
JP NZ 0020	**0048**	**C2 20 00**	Jump to start of program if A is pressed.
LD A,0F	**004B**	**3E 0F**	Load A with **0F** to produce a 'backward C'.
OUT (02),A	**004D**	**D3 02**	Output **0F** to port 2.
LD B,08	**004F**	**06 08**	Load B with 08 for a short DELAY ROUTINE.
DJNZ 0051	**0051**	**10 FE**	**DJNZ** decrements register B to zero.
LD A,B9	**0053**	**3E B9**	Load the accumulator with **B9** to produce 'C' in display 1.
OUT (02),A	**0055**	**D3 02**	Output **B9** to port 2.
IN A,(01)	**0057**	**DB 01**	Input the two switches to see if either is pressed.
BIT 6,A	**0059**	**CB 77**	Test BIT 6 to see if B is pressed.
JR NZ 0066	**005B**	**20 09**	Jump if button B is pressed.
BIT 7,A	**005D**	**CB 7F**	Test BIT 7 to see if button A is pressed.
JR Z,004B	**005F**	**28 EA**	Jump back to line 24 if not pressed and loop constantly.
LD A,B0	**0061**	**3E B0**	If button A is pressed, load the accumulator with **B0**.
OUT (02),A	**0063**	**D3 02**	Output **B0** to port 2 to give '1' on display ONE.
HALT	**0065**	**76**	The program HALTS. Reset by pressing reset button.
BIT 7,A	**0066**	**CB 7F**	Test bit 7 to see if button A is also pressed.
JR Z,0074	**0068**	**28 0A**	Jump if button A is not pressed.
LD A,06	**006A**	**3E 06**	Load Accumulator with **06**
OUT (02),A	**006C**	**D3 02**	Output **06** to get '1' on display two.
LD A,B0	**006E**	**3E B0**	Load accumulator **B0**.
OUT (02),A	**0070**	**D3 02**	Output **B0** to port 2 to get '1' on display ONE.
JR 006A	**0072**	**18 F6**	Jump back to display 1's on both displays. Keep looping.
LD A,06	**0074**	**3E 06**	Load the accumulator with **06**.
OUT (02),A	**0076**	**D3 02**	Output **06** to port 2.
HALT	**0078**	**76**	Halt. Press reset button to reset game.

QUICK DRAW

Quick Draw is located at 0020 and this is addressed by switching ON the second lowest switch thus:



Quick Draw is a reaction game for two players. Player 'one' uses button A and player 'two' button B.

The game is played on the two 7-segment displays and the program starts by illuminating segments around the two displays. Then the perimeter of the two displays illuminate.

The first player to press his button is the winner and this is shown by a '1' appearing in the appropriate display.

If both players press at the same time, both displays illuminate.

If a player 'beats the gun', the game resets.

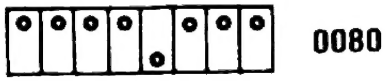
Press the reset button to start a new game.

Data Bytes at 00F5:

01
02
04
08
88
90
A0
81

RUNNING NAMES

To access this program, switch 8 must be ON. This will produce address-value 0080. Do not turn on switch 80 as this will produce 0800! Once the program has started, the switches can be turned OFF or set to the value necessary to access the name you want to appear on the screen.



Running names is a program which you use soon after the Microcomp has been completed.

It displays a message saying the builder of the project is YOU!

To do this we have included a list of about 30 names and these are accessed by loading the input port with a particular value, once the program is running.

Hopefully your name is amongst the list, but if not, there are a few general names at the end of the table to cover those excluded. Names containing M and W have been left out due to the difficulty in displaying them on the 7-segment displays. But for the majority, a name can be added to the message to add a personal flavour to the project.

The main program consists of 4 different sections. The first produces the message: "3-Chip uP built by". The second looks at the list of names and counts the FF's separating the names. It compares this with the value set on the input switches and displays the chosen name.

Part 3 of the program flashes 'C' on the screen to represent copyright and the 4th part of the program produces the date: **1985.**

The letters running across the displays are produced by a sub-routine which is used for the first, second and fourth parts of the program.

This sub-routine picks up the first two bytes in the table and displays them on the two displays. When the

RUNNING NAMES:

MAIN PROGRAM:

LD IX 0100	**0080**	**DD 21 00 01**	The IX register points to the start of the byte table
LD HL 008A	**0084**	**21 8A 00**	The HL register provides a return address for the sub-routine.
JP 00D0	**0087**	**C3 D0 00**	Jump to the RUNNING LETTER sub-routine
LD C,00	**008A**	**0E 00**	'C' is our COUNT register and is compared with an input value
LD IX 0114	**008C**	**DD 21 14 01**	IX is loaded with the start of the NAMES table
IN A,(01)	**0090**	**DB 01**	Input the value on the switches, to the accumulator
CP 00	**0092**	**FE 00**	If the input value is 00, the program increments to line 8 and
JR Z 00A9	**0094**	**28 13**	the micro jumps to line 20. If input value is NOT zero, to to 9.
LD D,A	**0096**	**57**	The input value is SAVED by loading it into register D.
LD A,(IX	**0097**	**DD 7E 00**	The data byte pointed to by the IX register is loaded into A.
CP FF	**009A**	**FE FF**	The accumulator is compared with FF to detect end of name.
JR Z 00A2	**009C**	**28 04**	If end of name is reached, the program jumps to line 15.
INC IX	**009E**	**DD 23**	If end of name not reached, INC IX and jump to line 10, where
JR 0097	**00A0**	**18 F5**	the next byte is loaded into A and compared with FF.
INC C	**00A2**	**0C**	The C register is incremented, indicating end of word.
LD A,C	**00A3**	**79**	Load C into the accumulator
CP D	**00A4**	**BA**	Compare accumulator with D to see if word has been located
JR NZ,009E	**00A5**	**20 F7**	Jump if word is not found
JR 00AB	**00A7**	**10 02**	Jump OVER line 20
DEC IX	**00A9**	**DD 2B**	This line only applies to the first word in the list.
LD HL,00B3	**00AB**	**21 B3 00**	Load HL with the return address for the sub-routine
INC IX	**00AE**	**DD 23**	Increment IX for the first letter of the name.
JP 00D0	**00B0**	**C3 D0 00**	Jump to the LETTER RUNNING routine and display name
LD C,08	**00B3**	**0E 08**	The C register is used to count 'COPYRIGHT' flashes
LD A,58	**00B5**	**3E 58**	Load accumulator with **58** to produce letter 'C' on display.
OUT (02),A	**00B7**	**D3 02**	Output 58 to port 2
DJNZ	**00B9**	**10 FE**	C remains ON for 256 loops of **DJNZ** (B register).
LD A,00	**00BB**	**3E 00**	Accumulator is loaded with zero
OUT (02),A	**00BD**	**D3 02**	Zero is outputted to turn OFF 'C'.
DJNZ	**00BF**	**10 FE**	Display is OFF for 256 loops of **DJNZ**.
DEC C	**00C1**	**0D**	The ON-OFF count register (RegisterC) is decremented.
JR NZ 00B5	**00C2**	**20 F1**	ON-OFF effect is repeated 8 times
LD IX,01F8	**00C4**	**DD 21 F8 01**	Register IX is loaded with 01F8, data for '1985'
LD HL,0080	**00C8**	**21 80 00**	Register HL is loaded with return address (re-start address)
JP 00D0	**00CB**	**C3 D0 00**	Program jumps to RUNNING LETTER routine.

RUNNING LETTER ROUTINE: - sub routine

LD C,0B	**00D0**	**0E 0B**	Each letter appears **0B** times (11 times)
LD A,(IX +00)	**00D2**	**DD 7E 00**	Load accumulator with byte pointed to by IX
SET 7,A	**00D5**	**CB FF**	SET bit 7, to turn on left-hand display
OUT (02),A	**00D6**	**D3 02**	The accumulator is outputted to port 2.
LD B,20	**00D7**	**06 20**	Load B with 20 (for 32 loops of **DJNZ**) for time delay.
DJNZ	**00D9**	**10 FE**	Perform 32 loops of decrementing register B.
LD A,(IX + 01)	**00DB**	**DD 7E 01**	Load the accumulator with next data byte in table.
OUT (02),A	**00DD**	**D3 02**	Output the accumulator to port 2
LD B,20	**00E0**	**06 20**	Load B with a value of 20 (32 in decimal)
DJNZ	**00E2**	**10 FE**	Decrement B 32 times
DEC C	**00E4**	**0D**	Decrement C
JR NZ,00D2	**00E6**	**20 E9**	If C is NOT zero, jump to line 2 and repeat **0B** times.
INC IX	**00E7**	**DD 23**	Increment the IX register
LD A,(IX + 01)	**00E9**	**DD 7E 01**	Load accumulator with next byte in table
CP FF	**00EB**	**FE FF**	Compare accumulator with FF.
JR NZ,00D2	**00EE**	**20 DE**	If accumulator is not FF, jump to start and shift letters across.
JP (HL)	**00F0**	**E9**	When FF is detected, micro jumps to address contained in HL.

clock speed is HIGH they will appear to be on at the same time. When the clock speed is LOW, they will produce a flickering effect.

The routine displays the letters for 0B cycles (11 cycles) and then looks at the next byte. If it is FF, the micro jumps back to the main program. If it is not FF, the sub-routine picks up the next byte and displays bytes 2 and 3 on the displays.

A table of names is situated at the end of the sub-routine, which is accessed by the main program and used by the sub-routine.

TABLE OF NAMES:

```
3    4F
     40
C    39
H    76
I    06
P    73
     00
up   1C
P    73
     00
BUILT 7C 3E 06 38 78 00
BY    7C 6E 00
ANDY  77 37 5E 6E
      FF
BASIL 7C 77 6D 06 38
      FF
BERT  &c 79 33 78
      FF
BILL  7C 06 38 38
      FF
BOB   7C 3F 7C
      FF
BRUCE 7C 33 3E 39 79
      FF
CARL  39 77 33 38
      FF
CHARLES 39 76 77 33 38 79 33
        00 00 00 00 00 00 FF
CLIFF 39 38 06 71 71
      FF
CLIVE 39 38 06 3E 79
      FF
CRIS  39 33 06 6D
      FF
COLIN 39 3F 38 06 37
      FF
CRAIG 39 33 77 06 3D
      FF
DAVID 5E 77 3E 06 5E
      FF
DOUG  5E 3F 3E 3D
      FF
ED    79 3E
      FF
EVAN  79 3E 77 37
      FF
GEORGE 3D 79 3F 33 3D 79
       FF 3D 38 79 37
       FF 3D 33 79 3D
       FF 06 77 37
       FF 1E 3F 76 37
       FF 73 77 78
       FF
GLEN  3D 38 79 37
      FF
GREG  3D 33 79 3D
      FF
IAN   06 77 37
      FF
JOHN  1E 3F 76 37
      FF
PAT   73 77 78
      FF
PETER 73 79 78 79 33
      FF
PHILIP 73 76 06 73
       FF 33 77 38 73 76
       FF 33 3F 6E
       FF 6D 39 3F 78
       78 FF
RALPH 33 77 38 73 76
      FF 33 3F 6E
      FF 6D 39 3F 78
      78 FF
ROY   33 3F 6E
      FF 6D 39 3F 78
      78 FF
SCOTT 6D 39 3F 78 78
      FF
STAN  6D 78 77 37
      FF 78 3F 37 6E
      FF 38 06 78 78 38 79
      00 3F 38 5E
      00 73 33 3F
      FF 06 77 37
      FF
TONY  78 3F 37 6E
      FF 38 06 78 78 38 79
      00 3F 38 5E
      00 73 33 3F
      FF 06 77 37
      FF
LITTLE 38 06 78 78 38 79
       00 3F 38 5E
       00 73 33 3F
       FF 06 77 37
       FF
OL I  38 00 06 FF
      ? 53
      ? 53
      ? 53
      FF
- 40
- 40
- 40
GUESS 3D 3E 79 6D 6D
      40
      40
      40
      FF
AN    77 37
      00
OLD   3F 38 5E
      00
PRO   73 33 3F
      FF
1     06
9     6F
8     7F
5     6D
      00
      00
      FF
```

The list of names in the table and the corresponding Hex value which must be placed on the input switches. If '8' is on the input, the message will read 'ENTER INPUT VALUE'.

1	ANDY
2	BASIL
3	BERT
4	BOB
5	BRUCE
6	CARL
7	CHARLES
8	ENTER INPUT VALUE
9	CLIFF
A	CLIVE
B	CRIS
C	COLIN
D	CRAIG
E	DAVID
F	DOUG
10	ED
11	EVEN
12	GEORGE
13	GLEN
14	GREG
15	IAN
16	JOHN
17	PAT
18	PETER
19	PHILIP
1A	RALPH
1B	ROY
1C	SCOTT
1D	STAN
1E	TONY
1F	LITTLE 'OL I
20	? ? ?
21	- - GUESS - -
22	AN OLD PRO

NUMBERS AND LETTERS

To produce numbers and letters on the displays, you cannot load a data value of **01** and hope to get the figure '1' on the screen. You will get segment 'a' illuminated. This means the hex value of the required segments must be added together to achieve the required figure.

For example, to produce the figure '1', we must turn on segments 'b' and 'c'. The hex value for 'b' is 02 and for 'c' it is 04. Add these together to get 06. To create the figure '2' on the screen, we must illuminate segments a, b, d, e and g. The hex values for these are: 01, 02, 08, 10 and 40. Adding these together we get 5B.

This process has been continued for the alphabet and numbers as shown in the following table.

Some of the letters are hard to create on a 7-segment display and the closest possible resemblance has been created.

A	77
B	7C
C	39
D	5E
E	79
F	71
G	3D
H	76
I	06
J	1E
K	75
L	38
M	55
N	37
O	3F
P	73
Q	2F
R	33
S	6D
T	70
U	3E
V	1C
W	1D
X	64
Y	6E
Z	1B
1	06
2	5B
3	4F
4	66
5	6D
6	7D
7	07
8	7F
9	67
0	3F

This table gives you the full alphabet and numbers, along with the Hex value needed to produce the character. Most of the letters will be quickly recognised with 'M' and 'W' having a bar over the character to indicate it is repeated again to create the letter.

LOOKING AT DATA

This program lets you look at data in the EPROM. This way you can check each of the programs we have listed.

The program is located at 0200 and is accessed by turning on switch '20'. Push reset to access the program. Page zero address 0000 will be displayed. To access page 1, 2, 3, 4, 5, 6 or 7, the appropriate switches at the input port must be switched ON.

For page 1, turn on switch 1. For page 2, turn on switch 2. For page 3, turn on switches 1 and 2. etc. Switches 8, 10, 20, 40, and 80 are masked OFF via the instruction at **206** and thus they do not affect the page-accessing.

The program is designed to loop around FF bytes and at page '2' the program is capable of reading itself!!

At page zero (or any other page) the program starts by displaying the address value. This will be shown with LOW BRIGHTNESS. Pushing button A will display the value of data at the address. This will be shown with FULL BRIGHTNESS.

Pushing button A again will advance to address 01 and pressing button A again will show the data at this address.

A fast-forward facility is provided by pushing button B when the address value is being displayed. This will enable you to fast-forward around a page to pick up a missed location.

You can select a different page number at any time and the correct data will be displayed.

This program is very handy for reading the contents of the EPROM and proving the data to be as stated.

The display values are generated from a byte table situated at the end of the program and is as follows:

BYTE TABLE at 0280:

0 = 3F
1 = 0B
2 = 5B
3 = 4F
4 = 66
5 = 6D
6 = 7D
7 = 07
8 = 7F
9 = 67
A = 77
B = 7C
C = 39
D = 5E
E = 79
F = 71

All too soon we have run out of space. There are lots more programs in the EPROM and these will be covered in the next issue.

When you buy a kit you will be able to access these programs and see how they work.

The Microcomp is designed to fit into a cassette case and be stored like a book. Hopefully you will be using it all the time and it won't see the book-shelf. I hope I have encouraged you sufficiently to buy one of the kits. I'm sure it'll be the best decision you will ever make.

```
LD C,00          200  0E 00     C is our TEST register. BIT's are SET or RESET in the program.
LD E,00          202  1E 00     Register E holds the count, from 00 to FF. Zeroed at start.
IN A,(01)        204  DB 01     The value on the switches is loaded into the accumulator.
AND 07           206  E6 07     The accumulator is ANDed with 7 - only 01, 02 & 04 detected.
LD D,A           208  57        The value on the switches (up to 07) is saved in 'D'.
LD A,E           209  7B        The COUNT REGISTER is loaded in the accumulator.
AND 0F           20A  E6 0F     AND 0F removes the 4 upper bits leaving the 4 lower bits.
LD HL 0280       20C  21 80 02  Load HL with the start of the BYTE TABLE.
ADD A,L          20F  85        ADD 80 to the accumulator.
LD L,A           210  6F        A new value for L is created (for later use).
LD A,00          211  3E 00     The accumulator is zeroed.
OUT (02),A       213  D3 02     The accumulator is outputted to port 2.
LD B,10          215  06 10     B is loaded with 10 (16 in decimal)
DJNZ 0217        217  10 FE     DJNZ A delay of 16 is created.
LD A,(HL)        219  7E        The accumulator is loaded with the value pointed
OUT (02),A       21A  D3 02          to by HL and outputted to port 2.
LD A,E           21C  7B        The count register is loaded into the accumulator.
RRA              21D  1F        The accumulator is rotated RIGHT. The 4 high bits
RRA              21E  1F             move down to the 4 lower places and are ANDed
RRA              21F  1F             with 0F.
RRA              220  1F
AND 0F           221  E6 0F     AND 0F removes the 4 upper bits
LD HL,0280       223  21 80 02  HL is loaded with 0280.
ADD A,L          226  85        The L register is ADDed to the accumulator.
LD L,A           227  6F        A new value for L is created.
LD A,00          228  3E 00     Zero the accumulator.
OUT (02),A       22A  D3 02     Output the accumulator to port 2.
LD B,10          22C  06 10     Load B with 10 for a delay routine.
DJNZ 022E        22E  10 FE     DJNZ for 16 loops.
LD A,(HL)        230  7E        Load the accumulator with the value POINTED TO by HL.
SET 7,A          231  CB FF     SET bit 7 of the accumulator to turn on display 1.
OUT (02),A       233  D3 02     Output the accumulator to port 2.
IN A,(01)        235  DB 01     Look at the input switches.
BIT 7,A          237  CB 7F     Test bit 7 to see if switch A is pressed.
JR Z 0243        239  28 08     JUMP if it is not pressed.
SET 1,C          23B  CB C9     SET bit 1 of the C register indicating A pressed.
BIT 2,C          23D  CB 51     Test bit 2 of register C to see if it '1' or '0'.
JR NZ 0204       23F  20 C3     If it is '1', jump to line 3. If it is zero, jump to next loop.
JR 024E          241  18 0B     Jump to start of loop '2'.
RES 2,C          243  CB 91     Reset bit 2 of register C.
BIT 6,A          245  CB 77     Test bit 6 to see if button B is pressed.
JR Z, 0204       247  28 BB     If it is not pressed, jump to line 3.
INC E .NOP. NOP  249  1C 00 00  Increment register E.
JR 0204          24C  18 B6     Jump to line 3.
LD HL,0280       24E  21 80 02  Load HL with start of byte table.
LD A,(DE)        251  1A        Load A with the data pointed to by DE.
AND 0F           252  E6 0F     And the accumulator with 0F.
ADD A,L          254  85        Add register L to the accumulator.
LD L,A           255  6F        Create a new value for L.
LD A,(HL)        256  7E        Load A with the data pointed to by HL.
OUT (02),A       257  D3 02     Output this data to port 2.
LD A,(DE)        259  1A        Load A with the data byte pointed to by DE.
RRA              25A  1F        Rotate the accumulator RIGHT so that the 4 high order bits
RRA              25B  1F             are shifted to the 4 lower positions.
RRA              25C  1F
RRA              25D  1F
AND 0F           25E  E6 0F     AND the accumulator with 0F to remove the 4 upper bits.
LD HL, 0280      260  21 80 02  Load HL with start of DATA TABLE.
ADD A,L          263  85        Add register L to the accumulator.
LD L,A           264  6F        Create a new value for L.
LD A,(HL)        265  7E        Load A with the value pointed to by HL.
SET 7,A          266  CB FF     SET bit 7 of the accumulator to turn on display 1.
OUT (02),A       268  D3 02     Output the accumulator to port 2.
IN A,(01)        26A  DB 01     Look at the switches.
BIT 7,A          26C  CB 7F     Detect if button A has been pressed.
JR Z 027B        26E  28 0B     JUMP if button A has not been pressed.
SET 2,C          270  CB d1     SET bit 2 of register C indicating button A pressed.
BIT 1,C          272  CB 49     Test bit 1 of register C to see if button A has been released.
JR NZ 024E       274  20 D8     Jump if bit 1 of register C is '1'.
INC E .NOP. NOP  276  1C 00 00  Increment register E.
JR 0204          279  18 89     Jump to loop 1.
RES 1,C          27B  CB 89     Reset bit 1 of register C.
JR 024E          27D  18 CF     JUMP to start of loop 2.
```

Z80 Machine Codes EXPLAINED Part 1

A **BRIEF** description for each Z-80 instruction:

ADC A,( ) . The value of the byte pointed to by the address in ( ) **plus the value of the carry flag**, is added to the accumulator.

ADC A,B . . The value of the B register **plus the value of the carry flag** is added to the accumulator.

ADC HL,BC . The value of the register pair BC **plus the value of the carry flag** , is added to the HL register pair.

ADD A,( ) . The byte at the address ( ) is added to the accumulator

ADD A,B . . The value of the B register is added to the accumulator

ADD HL,BC . The value of the BC register pair is added to the HL register pair.

ADD IX,BC . The value of the BC register pair is added to the index register.

AND ( ) . . The value of the byte pointed to by the address in ( ) is logically **AND**ed with the accumulator.

AND A . . . The accumulator is **AND**ed with itself.

AND B . . . The B register is **AND**ed with the accumulator.

BIT 0,( ) . . Bit 0 of the byte pointed to by the address in ( ) is tested and if found to be '0', the **ZERO FLAG** is set to '1'.

BIT 0,A . . . Bit 0 of the A register is tested and if it is '1' the zero flag is set to '0'.

CALL Addr . . The program is diverted to a sub-routine

CALL C Address The CALL will only be performed if the carry flag in the F register is '1'.

CALL M Address The CALL will only be performed if the S flag (sign flag) is negative.

CALL NC Address The CALL will only be performed if the NON-CARRY condition is present. i.e. carry flag '0'

CALL NZ Address The CALL will only be performed if a **NON-ZERO** condition is satisfied, i.e. the zero flag is '0'

CALL P Address The CALL will only be performed if the sign flag in the F register is positive. i.e S = 1.

CALL PE Address The CALL instruction will only be executed if the PARITY is EVEN. This means the P/V flag is SET (1)

CALL PO Address The CALL directive will only be executed if the PARITY is ODD. This means the P/V flag is reset (0)

CALL Z Address The CALL will only be executed if the zero flag is SET (1).

CCF . . . . . Complements the CARRY FLAG - reverses the condition of the carry flag.

CP ( ) . . . Compare the value of the byte pointed to by the address in ( ) with the accumulator.

CP A . . . . The accumulator is compared with itself.

CP B . . . . The value of the B register is compared with the accumulator.

CPD . . . . The contents of the memory location pointed to by the HL register pair is subtracted from the accumulator and the result discarded. Both HL and BC are decremented.

CPDR . . . . As above but instruction will terminate when BC = 0 or A = (HL).

CPI . . . . . The contents of the memory location pointed to by the HL register pair are compared with the contents of the accumulator. HL is then incremented and BC decremented.

CPIR . . . . Compare the contents of the address pointed to by the HL register pair with the accumulator. HL is then incremented and BC decremented until BC = 0. or if A = (HL).

CPL . . . . . Complement the accumulator. i.e. all 1's are changed to 0's etc.

DAA . . . . . Decimal Adjust the Accumulator. Produces one digit for the 4 least significant bits and one for the 4 most significant bits. The carry flag is set to '1' if an overflow occurs.

DEC ( ) . . . Decrement the contents of a memory location by one.

DEC A . . . . Decrement the contents of a CPU register by one.

DI . . . . . Disable a maskable interrupt signal.

DJNZ . . . . A conditional relative addressing jump. The contents of register B is firstly decremented. If the result is NOT ZERO, a jump, determined by the value of the displacement byte, will take place. If the result is zero, the next instruction will be executed.

EI . . . . . This one-byte instruction ENABLES the maskable interrupt function by setting the interrupt flip flops.

EX (SP)HL . The contents of the location addressed by the Stack Pointer are exchanged with the contents of the CPU register L. The contents of the H register are exchanged with the contents of the stack pointer plus one.

EX AF,AF . . The contents of the accumulator and status register are exchanged with the contents of the alternate accumulator and status register.

EX DE,HL . . Exchange the contents of DE and HL registers.

EXX . . . . Exchange the contents of the general purpose registers with corresponding alternate registers.

HALT . . . . CPU suspends operation and executes NOP's. It maintains memory refresh logic.

IM 0 . . . Sets interrupt mode 0

IM 1 . . . Program RESTARTS at 0038H when interrupted.

IM 2 . . . Sets interrupt mode 2

IN A,(C) . . Data (from the input port specified by the contents of register C) is loaded into register A.

INC (HL) . . The contents of a memory location are incremented by one

INC A . HL Increment register A (Or a register Pair e.g. BC DE etc.)

IND . . . Input from a port specified by the contents of register C. One byte of data is transferred to the memory location addressed by the contents of the HL register pair. The value in register B and HL will be decremented at the end of this instruction.

INDR . . . Same as IND except the instruction continues until register B reaches zero

INI . . . . Same as IND except the contents of HL register pair are incremented at the conclusion of the instruction.

INIR . . . Same as INI except the instruction repeats until register B reaches zero.

JP (HL) . . Jump to the address contained in register pair HL.

JP ADDR . . See CALL instructions

JP C,ADDR . A conditional relative addressing jump. See CALL for meaning of C, NC, NZ & Z.

LD (ADDR),A The contents of the accumulator are loaded into the address contained within the ( ).

LD (ADDR),BC The contents of C are loaded into the immediate address and B into the address plus one.

LD (BC),A . The contents of the accumulator are loaded into the location pointed to by the contents of BC.

LD ( ),A (IX + dis) The contents of A are loaded into the memory location obtained by adding a displacement value to the contents of the IX register.

LD A,(ADDR) Load the accumulator with the contents of the immediately specified memory address.

LD A,A . . . Load the data from one CPU register into another.

LD A,dd . . Load one byte of data into the CPU register specified.

LD BC,dd dd Load 2 bytes of data into the CPU register pair specified e.g. The first byte loads into C and the second byte into B.

LD A,I . . . Load the accumulator with the contents of the Interrupt Vector register.

LD A,R . . . Load the accumulator with the contents of the Memory Refresh register.

LD I,A . . . Load the Interrupt vector register, from the accumulator.

LD R,A . . . Load the Memory Refresh register with the contents of the accumulator.

LDD . . . . The contents of a memory location pointed to by the contents of the HL register pair are transferred to the location pointed to by the contents of the DE register pair. After the data has been transferred both HL and DE are decremented by one. Also the 'counter-register' pair BC is decreased by one.

LDDR . . . . Same as LDD except if contents of BC do not go to zero, the Program Counter will be decreased by a value of two and the instruction will be re-executed. The instruction will continue until the value in the register pair BC goes to zero.

LDI . . . . . Same as LDD except register pairs HL and DE are increased by a count of one.

LDIR . . . . Same as LDDR except register pairs HL and DE are incremented by one after the data has been transferred.

DESIGNED and PUBLISHED by COLIN MITCHELL at 35 Rosewarne Ave., CHELTENHAM, VICTORIA. 3192 (03) 584 2386